Journal of Autonomous Agents and Multi-Agent Systems manuscript No.
(will be inserted by the editor)

Can Bounded and Self-Interested Agents be Teammates?

Application to Planning in Ad Hoc Teams

Muthukumaran Chandrasekaran · Prashant Doshi · Yifeng Zeng · Yingke Chen

Received: date / Accepted: date

Abstract Planning for ad hoc teamwork is challenging because it involves agents collaborating without any prior coordination or communication. The focus is on principled methods for a single agent to cooperate with others. This motivates investigating the ad hoc teamwork problem in the context of self-interested decision-making frameworks. However, agents engaged in individual decision making in multiagent settings face the task of having to reason about other agents' actions, which may in turn involve reasoning about others. An established approximation that operationalizes this approach is to bound the infinite nesting from below by introducing level 0 models. For the purposes of this study, individual, self-interested decision making in multiagent settings is modeled using interactive dynamic influence diagrams (I-DID). These are graphical models with the benefit that they naturally offer a factored representation of the problem allowing agents to ascribe dynamic models to others and reason about them.

We demonstrate that an implication of bounded, finitely-nested reasoning by a self-interested agent is that we may not obtain optimal team solutions in cooperative settings if it is part of a team. We address this limitation by including models at level 0 whose solutions involve reinforcement learning. We show how the learning is integrated into planning in the context of I-DIDs. This facilitates optimal teammate behavior, and we demonstrate its applicability to ad hoc teamwork on several problem domains and configurations.

Muthukumaran Chandrasekaran
THINC Lab, University of Georgia, Athens, Georgia 30602, USA
E-mail: mkran@uga.edu

Prashant Doshi
THINC Lab, University of Georgia, Athens, Georgia 30602, USA
E-mail: pdoshi@cs.uga.edu

Yifeng Zeng
School of Computing, Teesside University, Middlesbrough, Tees Valley, UK TS13BA
E-mail: y.zeng@tees.ac.uk

Yingke Chen
Sichuan University, Chengdu, China
E-mail: cik1984@gmail.com

Keywords multiagent systems · ad hoc teamwork · sequential decision making and planning · reinforcement learning

1 Introduction

Ad hoc teamwork – also called impromptu or spontaneous teamwork [20] – involves a team of autonomous and artificial agents with differing beliefs and capabilities coming together in cooperation toward a common goal without any prior coordination or communication protocols [68]. While the problem pertains to teamwork, the challenge is to essentially build an autonomous agent whose decision making involves reasoning about interactions with other agents. This necessitates understanding how others in the ad hoc team could be behaving, how their behaviors change over time in extended interactions, and the implications of others' behaviors.

The preclusion of prior coordination and commonalities as much as possible severely challenges planning in ad hoc settings. For example, well-known cooperative planning frameworks such as the communicative multiagent team decision problem [62] and the decentralized partially observable Markov decision process (DEC-POMDP) [16] utilize centralized offline planning and distribution of local policies among agents that have common initial beliefs. This presumed commonality makes these frameworks implausible for use in ad hoc settings. This is unfortunate because much advance has been achieved in the last decade in scaling the planning as modeled by DEC-POMDPs to well-sized teams of agents situated in large problems [11,26,31,49,54,57].

A key focus of ad hoc teamwork is on how an agent should behave online as a spontaneous teammate, which informs previous approaches toward planning. This includes an algorithm for online planning in ad hoc teams [74], OPAT, that solves a series of stage games assuming that other agents are optimal. Albrecht and Ramamoorthy [8] model the uncertainty about other agents' types and construct a Harsanyi-Bayesian ad hoc game that is solved online using reinforcement learning. While these early methods take important steps, they assume that the physical states and actions of others are perfectly observable. These approaches and their outcomes are informing our understanding of how planning should be carried out in ad hoc situations, but they do not apply to domains such as robotics where the physical state is not perfectly observed due to say, sensory noise.

The focus on individual agents' behaviors motivates that we situate the problem in the context of individual decision-making frameworks. Team reward is then some additive function of the individual agent rewards [5,29,41,50,72]. In this regard, recognized frameworks such as the interactive POMDP (I-POMDP) [39], its online and graphical counterpart, interactive dynamic influence diagram (I-DID) [36], and I-POMDP Lite [44] take a decision-theoretic approach to planning and model the individual agent as being self interested. I-POMDPs and I-DIDs provide formal models that are sufficiently general to accommodate a wide spectrum of planning problems [32] including that of ad hoc coordination. These frameworks allow considerations of partial observability of the state and uncertainty over the models and types of other agents with minimal prior assumptions. Of course, this generality comes with the expense of increased computational complexity. Indeed, Albrecht and Ramamoorthy [8] note the suitability of these frameworks to the problem of ad hoc teamwork but find the complexity challenging. We ground the investigations presented in this article using I-DIDs because they naturally provide for a factored representation that is also

suited for online planning. Furthermore, recent advances on model equivalences [75] and use of streaming multi-processors [2] allow I-DIDs to move beyond toy-scale problems. [1]

Self-interested agents that reason about others' behaviors (as in I-DIDs) should consider the possibility that others could be reasoning about others' as well. This simple observation gives rise to complex epistemology that continues to receive much attention in various branches of game theory such as epistemic [13,22] and behavioral [25,24], as well as in multiagent systems [36,39,7,75]. Toward making the ensuing hierarchical belief systems computationally operational, varied effort has converged toward capping the belief system from below by introducing level-0 models. Wright and Lleyton-Brown [73] survey various level-0 models employed in behavioral game theory and propose new meta models.

Level-0 models in I-DIDs are single-agent DIDs that do not reason about others. Agents at a strategy level l are capable of modeling others at lower levels (up to $l-1$) only. Therefore, agents which are more strategic are capable of modeling others at deeper levels (i.e., all levels less their own strategy level l), but are always only boundedly optimal. As such, these agents could fail to predict the strategy of a more sophisticated opponent. We show that a consequence of this strategic boundedness is that the individual agent may not behave as an optimal teammate. Intuitively, this is because the true benefit of cooperative actions often hinges on others performing supporting actions, which by themselves may not be highly rewarding to the agent. This finding is one of the first insights into how a specific form of bounded rationality negatively impacts team planning.

To address this, we augment I-DIDs by additionally attributing a new type of level-0 model. This type distinguishes itself by utilizing reinforcement learning (RL) either online or in simulation to discover possible collaborative policies that a level-0 agent may execute. In essence, this allows an individual agent at a lower level to learn the existence of other agents that are more sophisticated, and consider them to be a part of the world dynamics.

In this article, we seek to shape the answer to a general question: *Could strategic agents planning and acting individually in a multiagent setting each using, say the boundedly-optimal I-DID, engage in comprehensive team behavior?* To the best of our knowledge this broad question remains open, and its answer is challenging. Toward this question, the contributions of this article are three-fold:

1. We demonstrate using examples that a boundedly-optimal and self-interested agent may not behave as a teammate despite sharing its payoff function with others. Consequently, these agents are generally ill suited for participation in ad hoc teams.
2. We show that true team behavior emerges when the reasoning ability at base levels is enhanced via learning. In this regard, we demonstrate globally optimal teammate solutions when agents are modeled in finitely-nested *augmented* I-DIDs where *traditional* I-DIDs fail. Consequently, this work is of significance because it may provide us a way of generating optimal teammate behavior in boundedly rational frameworks. Until now, these have been utilized for noncooperative settings, which provides a principled way to solving ad hoc teamwork problems.
3. We demonstrate the applicability of augmented I-DIDs to ad hoc settings and show its effectiveness for varying types of teammates.
   Specifically, ad hoc teamwork is studied in domains where agents do not observe their physical states nor actions of others perfectly, and neither do they communicate. The agents are Bayesian, self-interested and boundedly-optimal as defined previously. In reference to the categorization in Stone *et al.*, we assume that the agents share a common

[1] A GUI-based software tool called Netus is freely available from `http://tinyurl.com/mwrtlvg` for designing I-DIDs.

goal and receive identical reinforcements as they act simultaneously [68]. Analogous to Wu et al. [74], we experiment with multiple well-known cooperative domains exhibiting these settings. Subsequently, we implement a generalized version of OPAT that accounts for the partial observability and use that as a baseline for comparison with augmented I-DIDs.

This article is structured as follows. In Section 2, we analyze and differentiate the related work. In Section 3, we briefly review the framework of I-DIDs that underlies our work and discuss its relevance in the context of ad hoc settings. In Section 4, we illustrate the challenge faced by boundedly-rational agents planning and acting individually toward engaging in team behavior. We also present a new level-0 model with reasoning capabilities that are enhanced by utilizing reinforcement learning. We also present a revised method to solve the *augmented* I-DIDs. Furthermore, to handle computational explosion, we propose and illustrate a principled way to generate a reduced space of collaborative level-0 models and retain *top-K* of these models. Experimental results in Section 5 demonstrate the emergence of team behavior in I-DIDs and their applicability to ad hoc settings in three well-known cooperative ad hoc domains. Section 6 concludes this paper with a discussion and future lines of work. An appendix provides illustrations of I-DIDs for the various domains used in the experiments.

2 Related Work

Multiagent teamwork is a long standing and widely studied area of research. Relevant to the contributions in this article are the decision-theoretic frameworks that have significantly driven recent advancements in this area. Key among these are the multiagent (PO)MDP [19, 12], decentralized (PO)MDP [16] and their specializations such as the networked-distributed POMDP [46]. These frameworks model traditional teamwork as sequential decision making differing in the structure of the team plan (and whether the state is partially observable). All teammates are assumed to be identical in their capabilities, preferences and beliefs.

Recently, interest in spontaneously coordinating with others as part of an ad hoc (or impromptu) multiagent team is building, driven by applications such as robot soccer [21] and disaster response. Requirements for this type of coordination include disallowing any prior coordination or relying on commonalities because potential teammates may not be known in advance. These requirements defeat the traditional team planning frameworks and make the coordination problem challenging because team strategies cannot be determined a'priori [68].

Autonomy remains a key attribute, and agents are expected to learn and adapt to the behaviors of others on the fly and still function effectively as teammates [15,70]. Opponent modeling techniques – initially explored in game theory – are useful in learning the behavior of others from the interaction history as in fictitious play [23], rational learning [45] and case-based reasoning [38]. The focus on how an agent should behave online as an ad hoc teammate informs previous approaches toward the planning. This includes an algorithm for online planning in ad hoc teams (OPAT) [74] that solves a series of stage games assuming that other agents are optimal with the utility at each stage computed using Monte Carlo tree search. Albrecht and Ramamoorthy [6,8] generalize Bayesian games [43] to model the uncertainty about other agents' user-defined types and construct a Harsanyi-Bayesian ad hoc coordination game (HBA) that is solved online using learning. However, establishing common knowledge of the prior distribution over types to facilitate the solution of HBAs is problematic in ad hoc settings. Albrecht *et al.* [9] further analyze the convergence conditions

of various posterior belief formulations to understand how closely the user-defined type spaces must approximate the true type spaces in order for HBA to effectively solve its task. Continuing on this line of research, Albrecht *et al.* [10] also propose an algorithm to quantify the correctness of predefined type spaces. While these approaches take important steps, they assume that the physical states and actions of others are perfectly observable, which often may not apply.

Ad hoc teamwork has been studied from other perspectives as well such as first identifying the target task(s) before determining the strategy adopted by the teammates [20] because the agent may not always know beforehand the task it is expected to complete. Our work, like many of the existing studies in ad hoc teamwork [4,27,69], assumes that the agents are aware of the tasks they are supposed to execute and does not consider the problem of identifying it.

Methods for ad hoc planning in this article are presented in the context of I-DIDs, which differ from other frameworks such as DEC-POMDPs [65] and multiagent influence diagrams (MAIDs) [48], in that they take the perspective of an individual agent to sequential decision making in multiagent settings, and do not assume common knowledge of beliefs between multiple agents.

As expected, solving I-DIDs (and I-POMDPs) tends to be computationally complex as they acutely suffer from both the curses of dimensionality and history [36,39,60]. Given this complexity, a plethora of principled approaches have been proposed for scaling exact and approximate I-DID solutions [28,33–35,61,63,75,76]. These methods primarily focus on reducing the dimensionality of the state space by exploiting *behavioral equivalence* (BE) among models that prescribe identical behavior. Models that are BE are then clustered and a single representative model for each equivalence class is picked. For example, Doshi and Zeng [35] minimize the model space by updating only those models that lead to behaviorally distinct models at the next time step. This approach, called *discriminative model updates*, speeds up solutions of I-DIDs considerably.

Like I-DIDs (and I-POMDPs), Agmon *et al.* [3] model uncertainty in behavior of other agents using recursive modeling to achieve ad hoc teamwork, but do so in the context of a simultaneous action repeated game. However, the goal of the ad hoc agent here is to lead its teammates, whose behaviors are given, to a joint action that results in higher team utility compared to what could be achieved without its intervention. This work reassures our argument that the true benefit of cooperative actions often hinges on others performing supporting actions, which by themselves may not be highly rewarding to the agent. We take this to the next level and seek to achieve true team behavior despite the bounded rationality limitations on finitely nested hierarchical systems by introducing enhanced level-0 models.

3 Background: Interactive DIDs

The decision making processes for multiagent settings in a stochastic partially observable environment could be formalized as decentralized partially observable Markov decision processes (Dec-POMDPs), interactive POMDPs (I-POMDPs) or others [32]. Dec-POMDPs model cooperative agents as a team who share joint beliefs over states and a local policy is provided to each agent. I-POMDPs take the perspective of an individual agent operating in presence of other self-interested agents. They are suitable for both cooperative and agnostic settings.

I-DIDs are the graphical counterparts of I-POMDPs. Interactive influence diagrams (I-IDs) and their dynamic counterparts, I-DIDs, seek to explicitly model the structure that is

often present in real-world problems by decomposing the situation into chance and decision variables, and the dependencies between the variables. I-DIDs generalize dynamic IDs (DIDs), which are graphical counterparts of POMDPs, to multiagent settings in the same way that I-POMDPs generalize POMDPs. Analogous to DIDs, I-DIDs compactly represent the decision problem by mapping various variables into chance, decision and utility nodes. However, we consider more complex multiagent interactions that are extended over time. So, predictions about others' future actions must also be made by solving models that need to be updated as the agents act and observe. I-DIDs allow the explicit representation of other agents' models as the values of a special *model node*. Other agents' models and the original agent's beliefs over these models are then updated over time. Specifically, the update of the agent's belief over the models of others as the agents act and receive observations is denoted using a special link called the *model update link* that connects the model nodes between time steps.

Frameworks such as I-DIDs are significant in that they operationalize hierarchical belief systems for use in decision making, which have received extensive mathematical treatment in game theory [1,13,52]. They also contribute to a growing line of work on graphical models for multiagent decision making that includes multiagent influence diagrams (MAID) [47], and more recently, networks of influence diagrams (NID) [37]. MAIDs objectively analyze the game and efficiently compute the Nash equilibrium profile by exploiting the independence structure. NIDs extend MAIDs to include agents' uncertainty over the game being played and over models of other agents. However, MAIDs provide an analysis from an external viewpoint and the applicability of both is limited to static single play games. MAIDs do not allow us to define a distribution over non-equilibrium behaviors of other agents. In comparison, I-DIDs provide a way to exploit predicted non-equilibrium behavior. Thus, MAIDs and hence NIDs, are not amenable to modeling decision making in multiagent settings from an individual agent's perspective.

I-DIDs are naturally suited for planning in ad hoc multiagent settings because they take a decision-theoretic approach to planning and model the individual agent as being self-interested. A factored representation of the domain as utilized by I-DIDs promotes tractability in online planning [36]. Considerations of partial observability of the state and others' actions lead to uncertainty over the models of other agents with minimal prior assumptions. In this respect, I-DIDs allow us to maintain models of others, maintain distribution over these models, and update them over time. It explicitly models others as intentional agents and reasons about their behavior. Consequently, I-DIDs can quickly and spontaneously adapt to changing behaviors of teammates. We sketch I-DIDs below and refer readers to [75] for more details.

3.1 Representation

A traditional DID models sequential decision making for a single agent by linking a set of chance, decision and utility nodes over multiple time steps [71]. To consider multiagent interactions, I-DIDs introduce a new type of node called the *model node* (hexagonal node, $M_{j,l-1}$, in Fig. 1(a)) that represents how another agent j acts as the subject agent i reasons about its own decisions at level l. The model node contains as values the alternative computational models ascribed by i to the other agent j from the set of computable *intentional models* (and possibly *subintentional models*) of agent j at level $l-1$. A link from the chance node, S, to the model node, $M_{j,l-1}$, represents agent i's beliefs over j's models. Specifically, it is a probability distribution in the conditional probability table (CPT) of the chance

node, $Mod[M_j]$ (in Fig. 1(b)). An individual intentional model of j, $m_{j,l-1} = \langle b_{j,l-1}, \hat{\theta}_j \rangle$, where $b_{j,l-1}$ is the level $l-1$ belief, and $\hat{\theta}_j$ is the agent's *frame* encompassing the decision (rectangle), observation (oval) and utility (diamond) nodes. Each model, $m_{j,l-1}$, could be either a level $l-1$ I-DID or a DID at level 0. Solutions to the model are the predicted behavior of j and are encoded into the chance node, A_j, through a dashed link, called a *policy link*. Specifically, the dashed *policy link* which connects A_j to the model node $M_{j,l-1}$ represents the distribution over the other agent's actions given its model. In the absence of other agents, the model node and the chance node A_j vanish and I-DIDs collapse into traditional DIDs. Connecting A_j with other nodes in the I-DID structures how agent j's actions influence i's decision-making process. The CPD of the chance node, A_j, is a *multiplexer* that assumes the distribution of each of the action nodes (A_j^1, A_j^2) depending on the value of $Mod[M_j]$ (shown in Fig. 1(b)). In other words, when $Mod[M_j]$ takes the value $m_{j,l-1}^1$, A_j assumes the distribution A_j^1. Similarly, when $Mod[M_j]$ takes the value $m_{j,l-1}^2$, A_j assumes the distribution A_j^2 so on.

Expansion of an I-DID involves the update of the model node over time as indicated by the *model update link* - a dotted link from $M_{j,l-1}^t$ to $M_{j,l-1}^{t+1}$ in Fig. 1(a). As agent j acts and receives observations over time, its models should be updated. For each model $m_{j,l-1}^t$ at time t, its optimal solutions may include all actions and agent j may receive any of the possible observations. Consequently, the set of the updated models at $t+1$ contains up to $|\mathcal{M}_{j,l-1}^t||A_j||\Omega_j|$ models. Here, $|\mathcal{M}_{j,l-1}^t|$ is the number of models at time t, and $|A_j|$ and $|\Omega_j|$ the largest spaces of actions and observations respectively among all the models. The models differ in their initial beliefs updated using a configuration of action and observation. The CPT of $Mod[M_{j,l-1}^{t+1}]$ specifies the function, $\tau(b_{j,l-1}^t, a_j^t, o_j^{t+1}, b_{j,l-1}^{t+1})$ which is 1 if the belief $b_{j,l-1}^t$ in the model $m_{j,l-1}^t$ using the action a_j^t and observation o_j^{t+1} updates to $b_{j,l-1}^{t+1}$ in a model $m_{j,l-1}^{t+1}$; otherwise, it is 0. Note that the probability of j's possible observation, o_j^{t+1}, at time $t+1$, is obtained from the chance node O_j^{t+1}, which depending on the value of $Mod[M_j]$ assumes the distribution of the observation node in the lower level model. Similar to A_j, the CPD of O_j^{t+1} is also a multiplexer modulated by $Mod[M_j]$. Fig. 1(b) also clarifies the semantics of the policy link and the model update link, and shows how it can be represented using the traditional dependency links and chance nodes, and transform an I-DID into a traditional DID. For clarity, we elaborate an example I-DID for the well-studied multiagent grid meeting problem.

Example 1 (Level 1 I-DID) Figure 2 shows the I-DID for a level-1 agent i which considers two models of j at level-0 in the two-agent grid meeting problem. The two models, $m_{j,0}^{t,1}$ and $m_{j,0}^{t,2}$, differ in j's belief about the agents' location in the grid and are included in the model node $M_{j,0}^t$.

We show the update of $m_{j,0}^{t,1}$ and $m_{j,0}^{t,2}$ in Fig. 3. As agent j may receive one of four observations, eight updated models ($m_{j,0}^{t+1,1}, m_{j,0}^{t+1,1}, m_{j,0}^{t+1,2} \ldots m_{j,0}^{t+1,8}$) are generated in the model node $M_{j,0}^{t+1}$. The probability that j's updated model is, say $m_{j,0}^{t+1,1}$, depends on the probability of the j performing the action and receiving the observation that led to this model, and the prior distribution over j's model at time step t. Expansion of the I-DID over more time steps translates into repeating the steps of updating the set of models that form the values of the model node and adding the relationships between the chance nodes, as many times as there are model update links. Details of the model update link in the grid domain is shown in Fig. 3.

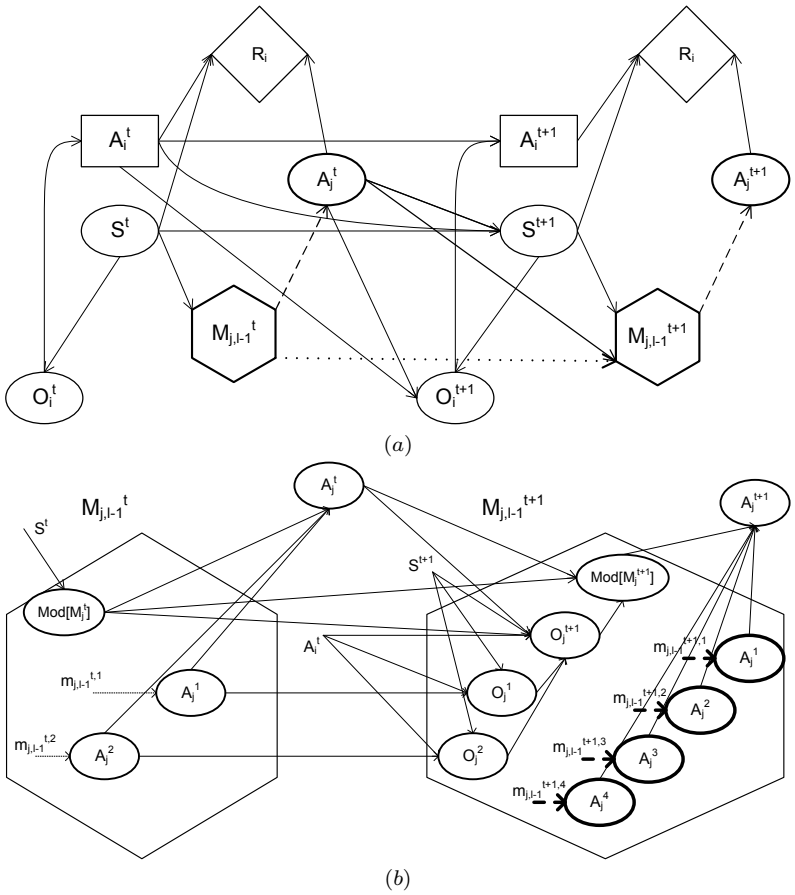

Fig. 1: (a) A generic two time-slice level l I-DID for agent i. The hexagons are the model nodes that contain the ascribed models of agent j. Each model includes j's belief – a probability distribution over chance node S^t in j's DID or I-DID – and a frame consisting of the rectangular action nodes, successive chance nodes and their CPTs, and the utility nodes, in j's two-time slice DID or I-DID. Policy links are marked as dash lines, while the model update link is marked as a dotted lines. The dotted model update link represents the update of j's models and the distribution over the models over time; (b) Implementation of the model update link using standard dependency links and chance nodes; e.g., two models, $m_{j,l-1}^{t,1}$ and $m_{j,l-1}^{t,2}$, are updated into four models (shown in bold) at time $t+1$.

3.2 Solution

A level l I-DID of agent i expanded over T time steps is solved in a bottom-up manner. To solve agent i's level l I-DID, all lower level $l-1$ models of agent j must be solved. Solution to a level $l-1$ model, $m_{j,l-1}$, is j's policy that is a mapping from j's observations in O_j to the optimal decision in A_j given its belief, $b_{j,l-1}$. Subsequently, we may enter j's optimal decisions into the chance node, A_j, at each time step and expand j's models in

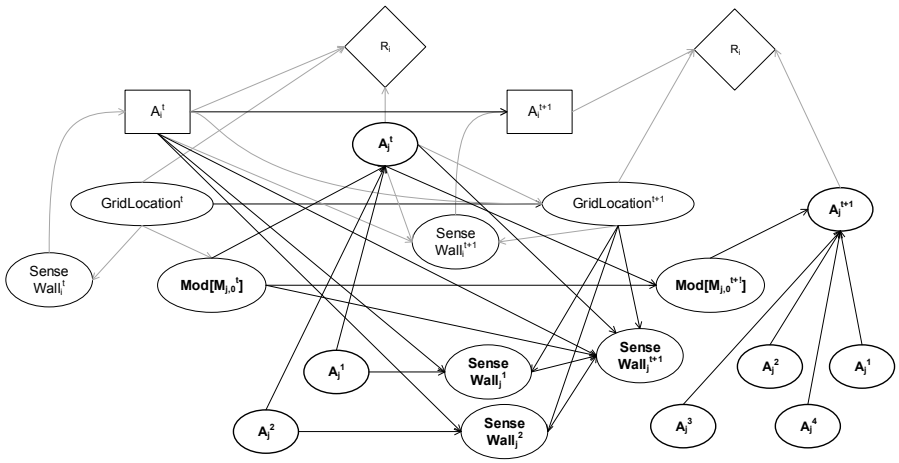

Fig. 2: A two time-slice level 1 I-DID for agent i in the grid problem. The model update link has been replaced by regular arcs in DIDs. The lower level models are solved to obtain the distributions for the chance action nodes.

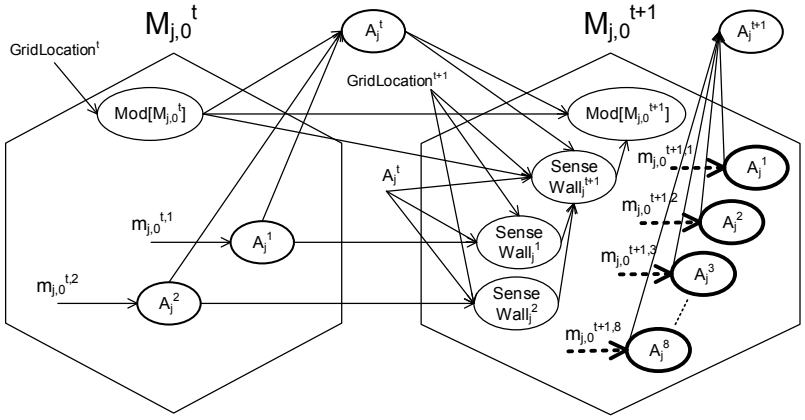

Fig. 3: Details of the model update link where two models are expanded into eight models in the new time step.

$Mod[M_{j,l-1}]$ corresponding to each pair of j's optimal action and observation. We perform this process for each of level $l-1$ models of j at each time step, and obtain the fully expanded level l model. We outline the algorithm for exactly solving I-DIDs in Fig. 4 [35].

The computational complexity of solving I-DIDs is mainly due to the exponential growth of lower $l-1$ j's models over time. Although the space of possible models is very large, not all models need to be considered in the model node. Models that are behaviorally equivalent (BE) [61] – whose behavioral predictions for the other agent are identical – could be pruned and a single representative model can be considered. This is because the solution of the

subject agent's I-DID is affected by the behavior of the other agent only; thus we need not distinguish between BE models. Let **PruneBehavioralEq** ($\mathcal{M}_{j,l-1}$) be the procedure that prunes BE models from $\mathcal{M}_{j,l-1}$ returning the representative models (line 6).

Note that lines 4-5 (in Fig. 4) solve level l-1 I-DIDs or DIDs and then supply the policies to level l I-DID. Due to the bounded rationality of level l-1 agents, the solutions lead to a suboptimal policy of agent j, which certainly compromises agent i's performance in the interactions particularly in a team setting. Also, note that the level 0 models are DIDs that do not involve learning. We will show in the coming sections that solving I-DIDs integrated with RL may generate the expected team behavior among agents i and j.

I-DID EXACT(level $l \geq 1$ I-DID or level 0 DID, horizon T)
Expansion Phase
1. **For** t **from** 0 **to** $T-1$ **do**
2. **If** $l \geq 1$ **then**
Populate $M^{t+1}_{j,l-1}$
3. **For each** m^t_j **in** $\mathcal{M}^t_{j,l-1}$ **do**
4. Recursively call algorithm with the $l-1$ I-DID (or DID) that represents m^t_j and horizon, $T-t$
5. Map the decision node of the solved I-DID (or DID), $OPT(m^t_j)$, to the corresponding chance node A_j
6. $\mathcal{M}^t_{j,l-1} \leftarrow$ **PruneBehavioralEq**($\mathcal{M}^t_{j,l-1}$)
7. **For each** m^t_j **in** $\mathcal{M}^t_{j,l-1}$ **do**
8. **For each** a_j **in** $OPT(m^t_j)$ **do**
9. **For each** o_j **in** O_j (part of m^t_j) **do**
10. Update j's belief, $b^{t+1}_j \leftarrow SE(b^t_j, a_j, o_j)$
11. $m^{t+1}_j \leftarrow$ New I-DID (or DID) with b^{t+1}_j
12. $\mathcal{M}^{t+1}_{j,l-1} \overset{\cup}{\leftarrow} \{m^{t+1}_j\}$
13. Add the model node, $M^{t+1}_{j,l-1}$, and the model update link
14. Add the chance, decision, and utility nodes for $t+1$ time slice and the dependency links between them
15. Establish the CPTs for each chance node and utility node
Solution Phase
16. **If** $l \geq 1$ **then**
17. Represent the model nodes, policy links and the model update links as in Fig. 1 to obtain the DID
18. Apply the standard look-ahead and backup method to solve the expanded DID

Fig. 4: Algorithm (originally introduced by Doshi *et al*. [34]) for exactly solving a *traditional* level $l \geq 1$ I-DID or level 0 DID expanded over T time steps.

4 Teamwork in Interactive DIDs

Ad hoc teamwork involves multiple agents working collaboratively in order to optimize the team reward. Each ad hoc agent in the team behaves according to a policy, which maps the agent's observation history or beliefs to the action(s) it should perform. We begin by showing that the finitely-nested hierarchy in I-DID does not facilitate ad hoc teamwork.

However, augmenting the traditional model space with models whose solution is obtained via reinforcement learning provides a way for team behavior to emerge.

4.1 Implausibility of Teamwork

As mentioned earlier, in this article, we seek to shape the answer to the general question: *Could intentional agents planning and acting individually in a multiagent setting each using, say the finitely-nested* I-DID*, engage in team behavior?* To the best of our knowledge, this specific question remains open.

We illustrate the challenge in observing team behavior when agents plan and act individually in a mutiagent setting using bounded optimalilty such as finitely-nested I-DIDs next. Figure 5 shows a team setting of a two-agent *grid meeting* problem [17]. An agent can detect the presence of a wall on its right (RW), left (LW) or the absence of it on both sides (NW). Given a specific observation, the agent may choose to either move in one of four cardinal directions – south (MS), north (MN), east (ME), or west (MW) – or stay in the same cell (ST). Each agent i or j moves in the grid and collects rewards as determined by the number indicated in the occupied cell. If they move to different cells, the agents get their own individual reward. However, if they move to the same cell allowing them to hold an ad hoc meeting, they will be rewarded with twice the sum of their individual rewards. Initial positions of the two agents are shown in color and we focus on their immediate actions.

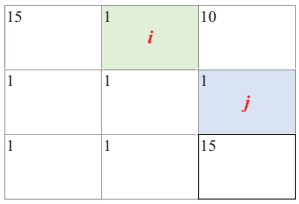

Fig. 5: Agents i and j in the grid meeting problem with the numbers corresponding to rewards. Each agent receives the sum of the rewards in the cells occupied by the agents.

Each agent, i or j, in a team receives the joint reward that is the sum of the individual reward proportional to the numbers in each cell occupied by the agents. The team's objective is to move to the same cell allowing them to hold a meeting and getting twice the sum of their individual rewards. Initial positions of the two agents are shown colored and we focus on their immediate actions.

If each agent is self interested and deliberates at its own level, agent i modeled at level 0 will choose to move left while a level-0 agent j chooses to move down. Each agent will occupy a cell with a reward of 15 and the whole team gets 30. Agent i modeled at level 1 and modeling j at level 0 thinks that j will move down, and its own best response to predicted j's behavior is to move left. Analogously, a level-1 agent j would choose to move down. A level 2 agent i will predict that a level-1 j moves down as mentioned previously, due to which it decides to move left. Analogously, a level-2 agent j continues to decide to move down. We may apply this reasoning inductively to conclude that level $l \geq 0$ agents i and j would move left and down, respectively, thereby earning a joint reward of 30. However,

the optimal team behavior in this setting is for i to move right and j to move up thereby obtaining a team reward of 40.

Established hierarchical classifications in game theory [1,13,52] and in multiagent systems [39] assert that at the lowest level (level 0), an intentional agent acts only on its own desires and the state of the world, with no understanding of the other agent's beliefs, preferences or capabilities. We formally define a *level-0 model* as: $m_{j,0} = \langle b_{j,0}, \hat{\theta}_j \rangle$, where $b_{j,0} \in \Delta(S)$ is the level-0 belief and $\hat{\theta}_j$ is agent j's *frame* encompassing the decision, observation and utility nodes in the DID. The level-0 belief is a probability distribution over the physical states only. In particular, agent j at level 0 does not explicitly model agent i; it acts without awareness of i in the common environment. Agents that are strategic model the other agents at levels up to one less than their own.

Clearly, this construction approximates the infinitely-nested belief system. One consequence of this bounded modeling is that decision making based on these finite hierarchical systems may not prescribe the optimal team behavior in cooperative settings. Observation 1 states this more formally:

Observation 1 *There exist cooperative multiagent settings in which boundedly-rational intentional agents each modeled using the finitely-nested I-DID (or I-POMDP) may not choose the jointly optimal behavior of working together as a team.*

Notice that an offline specification of level-0 models in cooperative settings is necessarily incomplete because level-0 semantics do not allow modeling others. This inhibits teamwork because level-0 models are not team aware and may not know the team task, as demonstrated in the grid meeting problem. However, the true benefit of cooperative actions often hinges on others performing supporting actions, which by themselves may not be highly rewarding to the agent. Thus, despite having an identical frame and solving the level 0 models optimally, the agent may not engage in optimal team behavior.

In general, this observation holds for cooperative settings where the self-maximizing level 0 models result in predictions that are not consistent with team behavior. Of course, settings may exist where the level-0 model's solution coincides with the policy of a teammate thereby leading to joint teamwork. Nevertheless, the significance of this observation is that we cannot rely on finitely-nested I-DIDs to generate optimal teammate policies.

We observe that team behavior is challenging in the context we study above because of the bounded rationality imposed by assuming the existence of a level 0. The boundedness precludes modeling others at the same level as one's own – as an equal teammate. However, at the same time, this imposition is, (a) motivated by reasons of computability, which allow us to operationalize such a paradigm; and (b) allows us to avoid some self-contradicting, and therefore impossible beliefs, which exist when infinite belief hierarchies are considered [18]. Consequently, this work is of significance because it may provide us a way of generating optimal team behavior in finitely-nested frameworks, which so far have been utilized for noncooperative settings; this provides a principled way to plan ad hoc teamwork.

4.2 Augmented Level 0 Models that Learn

We present a principled way to induce team behavior by enhancing the reasoning ability of lower-level models. While it is difficult to *a priori* discern the benefit of moving up for agent j in Fig. 5, it could be *experienced* by the agent. Specifically, it may explore moving in different directions including moving up and learn about its benefit from the ensuing, possibly indirect, team reward.

If we place agents in the cooperative problem domain or the simulated environment, an agent may come to *experience* the benefit of performing team actions. Subsequently, we may expect an agent to learn policies that are consistent with optimal teammate behavior because the corresponding actions provide large reinforcements. For example, given that agent i moves right in Fig. 5, j may choose to move up in its exploration, and thereby receive a large reinforcing reward. This insight motivates formulating level-0 models that utilize RL to generate the predicted policy for the modeled agent. Essentially, we expect that RL with its explorations could compensate for the lack of teamwork due to bounded reasoning in finitely-nested I-DIDs.

Because solutions of level-0 models are single-agent policies for the modeled agent only, we focus on the modeled agent's learning problem. However, rewards in the multiagent setting usually depend on actions of all agents due to which the other agent must be considered as well. The other agent's actions are a part of the environment and its presence hidden at level 0, thereby making the problem one of single-agent learning as opposed to one of multi-agent learning.

We augment the level-0 model space, with the augmented space denoted as $\mathcal{M}'_{j,0}$, by additionally attributing a new type of level 0 model to the other agent j: $m'_{j,0} = \langle b_{j,0}, \hat{\theta}'_j \rangle$, where $b_{j,0}$ is j's belief and $\hat{\theta}'_{j,0}$ is the frame of the learning model. The frame $\hat{\theta}'_{j,0}$ consists of the learning rate, α; a seed policy, $\pi_j^{(0)}$, of planning horizon T, which includes a fair amount of exploration; and the chance and utility nodes of the DID along with a candidate policy of agent i, which could be an arbitrary policy from i's policy space, Π_i, as agent i's actual behavior is not known. This permits a proper simulation of the multiagent environment.

This type of model $m'_{j,0}$ differs from a traditional DID based level-0 model in the aspect that $m'_{j,0}$ does not pre-specify the planning process of how agent j optimizes its decisions, but allows j to learn an optimal policy with the learning rate either online or in a simulated setting. Different models of agent j differ not only in their learning rates and seed policies, but also in i's candidate policy that is used. In principle, while the learning rate and seed policies may be held fixed, j's model space could be as large as i's policy space. Consequently, our augmented model space becomes extremely large.

4.3 Model-Free Learning: Generalized MCESP

Learning has been used toward decision making in both single- and multi-agent settings. Both model-based [30] and model-free [51,53,59] learning approaches exist for solving POMDPs. Banerjee *et al.* [14] utilized a distributed RL approach solving finite horizon DEC-POMDPs. Recently, Ng *et al.* [56] incorporated Bayesian model learning in the context of I-POMDPs where adversarial agents learn the transition and observation probabilities by utilizing parametric forms of transition and observation functions, and augmenting the interactive states with possible parameters of these functions.

A survey [58] classifies methods for multiagent learning into two broad categories: team learning where a single learner discovers joint policies and concurrent learning where multiple simultaneous learners are used. These approaches assume prior knowledge about the existence of other agents in the environment. In stark contrast to both forms of multiagent learning, our requirement is individual learning by an agent in a multiagent setting. This is a perspectivistic approach of multiagent learning that is understudied. Other than Ng *et al.*'s Bayes-adaptive I-POMDPs mentioned in the previous paragraph [56], we are unaware of

other methods for such learning in a partially-observable environment; however, this method is not model free.

Because the setting in which the learning takes place is partially observable, RL approaches that compute a table of values for state-action pairs such as traditional Q-learning do not apply. We adapt Perkin's Monte Carlo Exploring Starts for POMDPs (MCESP) [59], which has been shown to learn good policies in fewer iterations while making no prior assumptions about the agent's models in order to achieve convergence. MCESP maintains a Q table indexed by observation o_j and action a_j that gives the value of following a policy π_j except when observation o_j is obtained at which point action a_j that is different from the action prescribed by the policy π_j is performed. An agent's policy in MCESP maps the most recent observation to actions over the T time horizon, which is an approximation.

We generalize MCESP so that observation histories of length up to T, denoted as $\mathbf{o}$, are mapped to actions. A table entry, $Q^{\pi_j}_{\mathbf{o},a}$, is updated over every simulated trajectory of agent j, $\tau = \{\ \phi, a_j^0, r_j^0, o_j^1, a_j^1, r_j^1, \cdots, o_j^{T-1}, a_j^{T-1}, r_j^{T-1}, o_j^T\ \}$, where r_j is the team reward received and ϕ denotes no observation. Specifically, the $Q^{\pi_j}_{\mathbf{o},a}$ value is updated as:

$$Q^{\pi_j}_{\mathbf{o},a} \leftarrow (1-\alpha) Q^{\pi_j}_{\mathbf{o},a} + \alpha R_{post-\mathbf{o}}(\tau) \tag{1}$$

where α is the learning rate and $R_{post-\mathbf{o}}(\tau)$ is the sum of rewards of a portion of the observation-action sequence, τ, following the first occurrence of $\mathbf{o}$ in τ say at t':

$$R_{post-\mathbf{o}}(\tau) = \sum_{t=t'}^{T-1} \gamma^t r_j^t$$

where $\gamma \in [0, 1)$ is the discount factor. Alternate policies are considered by perturbing the action for randomly selected observation histories. As such, MCESP is analogous to policy iteration cf. the value iteration analogy of methods such as Q-learning.

Level-0 agent j learns its policy where the ad hoc agent i's actions are implicit in the environment. In other words, agent j needs to reason about the unknown behavior of i as it learns a level-0 policy using the generalized MCESP. Agent j considers the entire policy space of the ad hoc agent i Π_i and a fixed policy of i, $\pi_i (\in \Pi_i)$, results in one learned j's policy, π_j.

We show the algorithm for solving the ad hoc agent's teammate's level-0 models using the generalized MCESP in Fig. 6. The algorithm takes as input teammate j's model (with seed policy $\pi_j^{(0)}$) whose solution is needed and the fixed policy of the ad hoc agent i, which becomes a part of the environment. We note that by assuming a randomly initialized seed policy, we are able to achieve a fair amount of exploration because we now allow the agent j to learn a sub-optimal *teammate* policy $\dot{\pi}_j$ and consequently take sub-optimal *teammate* actions with a non-zero probability in response to the fixed policy π_i. We repeatedly obtain a trajectory τ of length T either by running the agent online or simulating the environment by sampling the states, actions and observations from the appropriate CPTs (lines 5-10). The trajectory is used in evaluating the value of the current transformed policy under consideration π_j of agent j (line 11). Initially, we utilize the seed policy contained in agent j's model. If another action a' for the observation sequence $\mathbf{o}$ is optimal after evaluating both the original and the transformed policy for the same number of $k > 0$ sample trajectories, we update π_j to conditionally use this action otherwise the policy remains unchanged (lines 12-13). This is followed by generating a perturbed policy in the neighborhood of the previous one (line 14), and the evaluation cycle is repeated. If the perturbation for every $(\mathbf{o}, a)$ pair is discarded, we may terminate the iterations returning the current policy and its action-value. For convenience, we denote the action-value $Q^{\pi_j}_{\mathbf{o},\pi_j(\mathbf{o})}$ more simply as Q^{π_j}.

RL FOR LEVEL 0 MODEL (j's model $m'_{j,0}$, i's policy π_i, T)

1. Sample the initial state s from $b_{j,0}$ in j's model
2. Set current policy of j denoted as π_j using the seed policy in j's model
3. Set $\tau \leftarrow \{\phi\}$ (empty observation)
4. **Repeat**
5. **For** t = 0 to $T-1$ **do**
6. Obtain i's action from π_i and j's action, a_j^t, from current policy of j using observation history
7. Obtain the next state, s', either by performing the actions or sampling
8. Obtain team reward, r_j^t, using state and joint actions
9. Obtain j's observation, o_j^{t+1}, using next state and joint actions
10. Generate trajectory, $\tau \leftarrow \tau \cup \{a_j^t, r_j^t, o_j^{t+1}\}$
11. Update $Q^{\pi_j}_{\mathbf{o},a}$ according to Eq. 1
12. **If** $\max_{a'} Q^{\pi_j}_{\mathbf{o},a'} > Q^{\pi_j}_{\mathbf{o},\pi_j(\mathbf{o})}$ and $Q^{\pi_j}_{\mathbf{o},a'}, Q^{\pi_j}_{\mathbf{o},\pi_j(\mathbf{o})}$ were both updated using k trajectories
13. $\pi_j(\mathbf{o}) \leftarrow a'$
14. Perturb an action a in π_j for some $\mathbf{o}$
15. **Until** termination condition is met
16. **Return** π_j and $Q^{\pi_j}_{\mathbf{o},\pi_j(\mathbf{o})}$

Fig. 6: Algorithm for learning agent j's policies when modeled at level 0.

Let $\mathcal{T}_j = \{tr_j\}$ be a collection of perturbations each of which maps a policy to another policy such that for some observation sequence another action will be performed. The following lemma shows that the generalized MCESP should yield a policy for agent j that is locally optimal in the limit. The lemma formalizes an observation made in Perkins [59].

Lemma 1 (Local optimality) *Let $\pi_j^{(1)}, \pi_j^{(2)}, \pi_j^{(3)}, \ldots, \dot{\pi}_j^{(n)}$ be the sequence of j's policies incrementally produced in* RL FOR LEVEL 0 MODEL *(m_j^0,π_i, T) where $\pi_j^{(c+1)} = \mathcal{T}_j(\pi_j^{(c)})$, $0 \leq c \leq n-1$. Then, in the limit of $k \rightarrow \infty$ we have,*

1. *The expected utility of each policy in the sequence is strictly greater than its predecessor, i.e.,*
$$Q^{\pi_j^{(c+1)}} > Q^{\pi_j^{(c)}} \quad \textit{for} \quad 0 \leq c \leq n-1$$
2. *The final policy $\dot{\pi}_j^{(n)}$ returned by the algorithm is locally optimal, i.e.,*
$$\neg\exists tr_j \in \mathcal{T}_j \quad Q^{tr_j(\dot{\pi}_j^{(n)})} > Q^{\dot{\pi}_j^{(n)}}$$

Proof As k approaches ∞, the estimated action values $Q^{\pi_j}_{\mathbf{o},a}$ for all $(\mathbf{o}, a)$ pairs as computed by Eq. 1 almost surely approaches the theoretical expected action-value by the strong law of large numbers – empirical distribution of post-$\mathbf{o}$ rewards obtained from the collection of k trajectories is identical to the exact distribution in the limit of $k \rightarrow \infty$ because observations and actions of j in the trajectories are i.i.d. given a fixed π_i.

Assertion (1) then follows from line 12 of Fig. 6 because updated policy in line 13 will always have an expected action-value that is greater than the action-value of the previous policy.

Let $\dot{\pi}_j^{(n)}$ be j's policy returned by the algorithm. According to the condition for termination, no other policy obtained by applying any perturbation in $\mathcal{T}_j$ that explores a different action for any single observation sequence yields a higher action-value. Therefore, $\dot{\pi}_j^{(n)}$ has the highest value in the space of neighboring policies as obtained by applying each perturbation in $\mathcal{T}_j$ to $\dot{\pi}_j^{(n)}$. Assertion (2) now follows. □

Proposition 1 below is derived easily given the lemma above. It states that the policy output from RL is of value that is at least as good as the seed policy that is input to the algorithm in the limit of $k \rightarrow \infty$.

Proposition 1 *Let $\dot{\pi}_j^{(n)}$ be agent j's policy returned by* RL FOR LEVEL 0 MODEL *(m_j^0,π_i, T) and $\pi_j^{(0)}$ be the seed policy in m_j^0. Then, as $k \rightarrow \infty$ the following holds in the limit:*

$$Q^{\dot{\pi}_j^{(n)}} \geq Q^{\pi_j^{(0)}} \tag{2}$$

Proof Consider the case where $\pi_j^{(0)}$ is locally optimal. In other words, no perturbation in $\mathcal{T}_j$ results in a policy whose action-value is greater than that of $\pi_j^{(0)}$. It follows that in the limit of $k \rightarrow \infty$, RL for Level 0 Model will return $\pi_j^{(0)}$. Therefore, Eq. 2 holds trivially.

Otherwise, if $\pi_j^{(0)}$ is not locally optimal then Lemma 1 establishes that the algorithm will incrementally produce a sequence of policies each of which improves in value over the previous one with the sequence culminating in $\dot{\pi}_j^{(n)}$ that is locally optimal. As the 'greater than in value' relation is obviously transitive, Eq. 2 holds in this case. □

Though the above mentioned theoretical results are established at the limit, these should begin to obtain in practice as the number of trajectories becomes sufficiently large.

As we mentioned previously, agent j's level 0 model space is inclusive of i's policies space $\mathit{\Pi}_i$. As the space of i's policy becomes large particularly for a large planning horizon, it is intractable for j to learn a policy for all i's candidate policies. In addition, considering that few of i's policies are actually collaborative we formulate a principled way to reduce the full space to those policies of i, denoted as $\hat{\mathit{\Pi}}_i$, that could be collaborative.

Our approach is as follows. We begin by picking a random initial policy of i, $\pi_i^{(0)}$, and using it in the frame of a new model of j. We apply generalized MCESP to this frame to obtain agent j's locally-optimal policy $\dot{\pi}_j^{(n)}$ where the seed $\pi_j^{(0)}$ was identical to $\pi_i^{(0)}$. Next, both the initial policy of j used by MCESP and i's policy is set to $\dot{\pi}_j^{(n)}$. MCESP then checks for the neighbors of $\dot{\pi}_j^{(n)}$, which improve on the joint utility of ($\dot{\pi}_j^{(n)}$,π_i ($= \dot{\pi}_j^{(n)}$)). If successful, an improved neighboring policy, say $\dot{\pi}_j^{(m)}$, is returned. Proposition 1 ensures that the utility of $\dot{\pi}_j^{(m)}$ is greater than or equal to $\dot{\pi}_j^{(n)}$ (used as the seed) for the same π_i ($= \dot{\pi}_j^{(n)}$).

We continue these iterations setting π_i as $\dot{\pi}_j^{(m)}$ and using $\dot{\pi}_j^{(m)}$ as the seed policy for j. MCESP cannot improve on $\dot{\pi}_j^{(m)}$ if $\dot{\pi}_j^{(m)}$ is also the local best response to $\pi_i = \dot{\pi}_j^{(m)}$. If this condition is satisfied, the procedure terminates. Otherwise, both $\dot{\pi}_j^{(n)}$ and $\dot{\pi}_j^{(m)}$ are added to the set of candidate predictions of level 0 behavior of j. The process may be restarted with a different random policy of the ad hoc agent i. We demonstrate this method on the 3×3 Grid domain in Fig. 7.

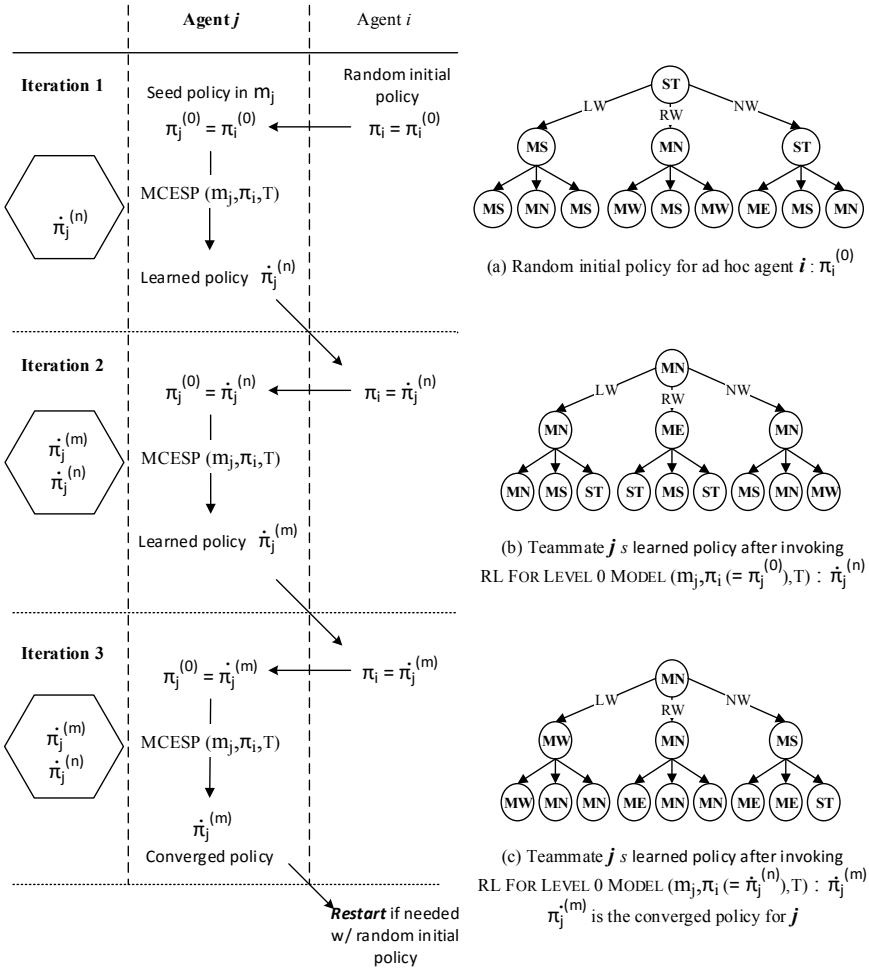

Fig. 7: We illustrate the iterative procedure to generate collaborative policies for lower-level teammate j in the 3x3 Grid. Hexagonal model node contains the set of candidate predictions of level-0 behavior of j. We start with (a) a random initial policy of ad hoc agent i $\pi_i^{(0)}$ in the frame of teammate j's model that is intrinsic in the environment; (b) execute generalized MCESP to generate a policy for agent j $\dot{\pi}_j^{(n)}$ obtained by invoking RL FOR LEVEL 0 MODEL on $(m_{j,0}, \pi_i)$ and we repeat this procedure until termination; (c) The learning converges to a fixed point $\dot{\pi}_j^{(m)}$ after which procedure may be restarted with another random initial policy for i.

We establish that the above iterative procedure represents a principled hill climb through the space of j's level-0 models for the team setting where the ad hoc agent i and its teammate j share a common reward function and initial belief over the state space. The team reward is equal to the sum of the individual agent rewards or its affine transformation, as is the case for the Grid domain of Fig. 5. Note that agents receive the team reward only. This is a transparent and typical way of deriving team rewards utilized by many cooperative problem domains. Proposition 2 also asserts that this procedure is devoid of oscillations and always terminates.

Proposition 2 (Hill Climb) *Let $\dot{\pi}_j^{(n)}, \dot{\pi}_j^{(m)}, \ldots, \dot{\pi}_j^{(q)}$ be j's policies produced by the above mentioned iterative procedure. Agent i's policies implicit in the frame begin with a random policy and are assigned the previously-obtained j's policy as described. Then, the following holds in the limit of $k \rightarrow \infty$:*

1. *Action-values reflecting the team utility always improve until the procedure terminates. Formally,*
$$Q^{\dot{\pi}_j^{(n)}} < Q^{\dot{\pi}_j^{(m)}} < \ldots < Q^{\dot{\pi}_j^{(q)}}$$
2. *The procedure will always terminate and the termination occurs at a fixed point obtained when* RL FOR LEVEL 0 MODEL *is invoked on $(\dot{\pi}_j^{(q)}, \pi_i(= \dot{\pi}_j^{(q)}), T)$ and it returns $\dot{\pi}_j^{(q)}$.*

Proof Run the iterative procedure on seed policies $(\pi_j^{(0)}(= \pi_i^{(0)}), \pi_i^{(0)})$. This involves invoking RL FOR LEVEL 0 MODEL with these seed policies, and let it return $\dot{\pi}_j^{(n)}$. Proposition 1 entails that the action-value reflecting the team utility $Q^{\dot{\pi}_j^{(n)}} > Q^{\pi_j^{(0)}}$. This implies a key observation that the contribution of $\dot{\pi}_j^{(n)}$ to the team reward is greater than the contribution of $\pi_j^{(0)}$ – former is a better plan for the team. Therefore, setting $\pi_i = \dot{\pi}_j^{(n)}$ in the next iteration will result in team utility of $(\dot{\pi}_j^{(n)}, \pi_i(= \dot{\pi}_j^{(n)}))$ that is strictly greater than that of $(\dot{\pi}_j^{(n)}, \pi_i^{(0)})$ because team utility is the sum of individual agent rewards (or its affine transformation). Furthermore, let $\dot{\pi}_j^{(m)}$ be returned from RL using the former as the seed. Again, Prop. 1 establishes that $Q^{\dot{\pi}_j^{(m)}} > Q^{\dot{\pi}_j^{(n)}}$ for the fixed $\pi_i = \pi_i^{(n)}$. Consequently, we assert that the action-value $Q^{\dot{\pi}_j^{(m)}}$ obtained from this iteration $> Q^{\dot{\pi}_j^{(n)}}$ as obtained from the previous iteration in which $\pi_i = \pi_j^{(0)}$. Assertion 1 then follows using same arguments for next iterations, until RL does not improve on the input.

Let RL FOR LEVEL 0 MODEL invoked on $(\dot{\pi}_j^{(q)}, \pi_i(= \dot{\pi}_j^{(q)}), T)$ as its input argument return $\dot{\pi}_j^{(q)}$ because no local pertubation results in a policy that improves on the seed. This is a fixed point for the iterative procedure because setting $\pi_i = \dot{\pi}_j^{(q)}$ in the next iteration results in the same input argument to RL as in the previous iteration. The procedure terminates when this fixed point is detected.

A potential oscillation may result if RL FOR LEVEL 0 MODEL invoked on $(\dot{\pi}_j^{(q)}, \pi_i(= \dot{\pi}_j^{(q)}), T)$ results in $\dot{\pi}_j^{(q-1)}$ – note that this formed the previous iteration's π_i. Consequently, π_i in the next iteration will also assume $\dot{\pi}_j^{(q-1)}$ thereby yielding a repeat of these two iterations that goes on infinitely. However, it is straightforward to show that such an oscillation is impossible. Let the RL on using input $(\dot{\pi}_j^{(q)}, \pi_i(= \dot{\pi}_j^{(q)}, T)$ yield $\dot{\pi}_j^{(q-1)}$. It follows from Prop. 1 that $Q^{\dot{\pi}_j^{(q-1)}} > Q^{\dot{\pi}_j^{(q)}}$ for the same π_i, which implies that $\dot{\pi}_j^{(q-1)}$ adds a larger individual reward to the team utility compared to $\dot{\pi}_j^{(q)}$. But this is a contradiction because

RL FOR LEVEL 0 MODEL invoked in the previous iteration on $(\dot{\pi}_j^{(q-1)}, \pi_i(=\dot{\pi}_j^{(q-1)}), T)$ yielded $\dot{\pi}_j^{(q)}$ implying that $\dot{\pi}_j^{(q)}$ contributes a larger reward than $\dot{\pi}_j^{(q-1)}$.

One or more optima that include Nash equilibria will always exist in team problems. More specifically, the global optimum is also a Pareto-optimal Nash equilibrium. □

Of course, as with any other local search procedure, our iterative hill climb may not yield the global optimum. Nevertheless, random restarts as suggested previously should assist.

4.4 Augmented I-DID Solutions

Solving augmented I-DIDs is similar to solving the traditional I-DIDs except that the candidate models of an agent at level 0 may be learning models. We show the revised algorithm in Fig. 8. When the level-0 model is a learning model, the algorithm invokes the method LEVEL 0 RL shown in Fig. 6. Otherwise, it follows the same procedure as shown in Fig. 4 to recursively solve the lower-level models.

While we consider a reduced space of agent i's policies in a principled way, and therefore agent j's learning models, we may further reduce agent j's policy space by heuristically keeping *top-K* policies of j, $K > 0$, in terms of their expected utilities (line 11 in Fig. 6). Observe that across models that differ in i's policy and with the same initial belief, team behavior(s) is guaranteed to generate the largest utility in a cooperative problem. This motivates focusing on models with higher utilities. Hence, the filtering of j's policy space may not compromise the quality of I-DID solutions at level 1. However, because MCESP may converge to a local optimum the resulting top-K policies are not guaranteed to include j's optimal collaborative policy in theory. Nonetheless, as our experiments demonstrate, we often obtain the optimal team behavior. As the number of optimal policies is unknown, we normally use a sufficiently large value of K.

Agent j's policy space may be additionally reduced because behaviorally equivalent models – learning and other models with identical solutions – will be clustered [75]. In summary, we take several steps that include both principled and heuristic to limit the negative impact of the increase in j's model space. Using a subset of i's policies preempts solving all j's models at level 0 while the top-K technique removes possibly non-collaborative policies.

5 Experiments

We conducted our experiments in two phases: First, we show that I-DIDs augmented with level-0 models that learn facilitate team behavior, which was previously implausible in traditional I-DID formulations. To this end, we begin by validating the policies generated by augmented I-DIDs by comparing with those obtained by a state-of-the-art DEC-POMDP formulation of popular cooperative domains. Next, we compare the expected utility of the subject agent's policies obtained by augmented I-DIDs with the values of the optimal team policies obtained using an exact DEC-POMDP approach. We establish the enhanced reasoning ability of level-0 models by demonstrating its benefit over traditional I-DIDs.

Second, we show the applicability of augmented I-DIDs to ad hoc teamwork in a setting similar to the one used by Wu *et al.* [74]. We begin by showing that in most cases, Augmented I-DIDs significantly outperform a known online ad hoc planner, OPAT, in terms of the solution quality by allowing for better adaptability. Note that we had to generalize OPAT to partially observable settings to facilitate a fair comparison. We analyze the significantly

AUGMENTED I-DID (level $l \geq 1$ I-DID or level 0 DID, T)

Expansion Phase

1. **For** t **from** 0 **to** $T-1$ **do**
2. **If** $l \geq 1$ **then**
   Populate $M^{t+1}_{j,l-1}$
   For each m^t_j **in** $\mathcal{M}^t_{j,l-1}$ **do**
3. **If** $l > 1$ **then**
4. Recursively call algorithm with the $l-1$ I-DID that represents m^t_j and the horizon, $T-t$
5. **else**
6. **If** the level 0 model is a learning model **then**
7. Solve using LEVEL 0 RL in Fig. 6 with horizon, $T-t$
8. **else**
9. Recursively call algorithm with level 0 DID and the horizon, $T-t$
10. Select top-K j's policies based on expected utility values given the same belief
11. The remaining steps of the expansion phase are the same as in Fig. 4.

Solution Phase

12. This is similar to the solution phase in Fig. 4.

Fig. 8: Algorithm for solving a level $l \geq 1$ I-DID or level-0 DID expanded over T time steps with $M'_{j,0}$ containing level 0 models that learn.

increased online run times for augmented I-DIDs over OPAT and identify its root cause – the overhead due to the learning method of MCESP – as an avenue for immediate future work.

Finally, our experiments provide strong evidence toward drawing the important conclusion that using RL to enhance the reasoning ability of level-0 models only is sufficient to induce teamwork by bounded and self-interested agents.

5.1 Problem Domains

We empirically evaluate the performance of AUGMENTED I-DID in different configurations of *three* well-known cooperative domains involving two agents, i and j: grid meeting (Grid) [17], box-pushing (BP) [64], and multi-access broadcast channel (MABC) [42]. We show I-DID formulations of each of these domains in the Appendix. First, we summarize the domains below:

Grid meeting domain (Grid). A two-agent 3×3 grid meeting (Grid) domain is illustrated in Fig. 5. The problem has 9 physical states for a single agent where each state refers to the agent's position in the grid. Therefore, for the agent at level 1 and above , the problem contains 81 states where each state refers to the joint position of both agents (i and j). The agents can detect the presence of a wall on its right (RW), left (LW) or the absence of it on both sides (NW). Given a specific observation, the agent may choose to either move in one of four cardinal directions – south (MS), north (MN), east (ME), or west (MW), or stay in the same cell (ST). Each agent's actions are assumed to have the expected result only 60% of the time. The agent may transition to a cell in any other direction or stay in the same cell each with a 10% probability. Movement into a wall returns the agent to its original state.

As each ad hoc agent moves in the grid, they collect rewards indicated by the numbers in the occupied cell. If they move to different cells, the agents get the sum of their individual rewards. If they move to the same cell allowing them to hold an ad hoc meeting, they will be rewarded with twice the sum of their individual rewards. A two-time slice I-DID for Grid is shown in the Appendix.

Box-pushing domain (BP). Fig. 9 illustrates the two-agent box-pushing (BP) domain in a 3×3 grid setting. In this domain, the two agents intend to push either a small or large box into the goal region. The small box can be moved by a single agent whereas the large one needs both agents to collaboratively push it. Each cell can host only one agent at any point in time. The problem motivates teamwork and agents' rewards are maximum when both of them cooperate and push the large box into the goal. Physical states represent the joint position and orientation of agents i and j. Additionally, there are two terminal states indicating that either the small box or the large box is at the goal. Hence, there are 50 states including the 2 terminal states in this domain. Each agent has 4 actions: turn left (TL), turn right (TR), move forward (MF) and stay (ST). An agent can move a box into the goal only when it is facing the box and performs the action MF. Each agent's actions are assumed to be successful 90% of the time and for the remaining 10%, it simply stays in place. To push the large box into the goal, both agents need to face north and move forward and by doing so, they achieve the largest reward of 100. The team is penalized 5 points if only one agent attempts to push the large box. The reward for pushing the small box into the goal region is 10 and the reward for being at any non-terminal state is -0.1 per agent to discourage excessive movement. Each agent is also equipped with sensors for observing its surroundings. It may receive one of 5 possible observations: empty field (EF), wall in front (WF), other agent (OA), small box in front (SF), or large box in front (LF). Appendix shows the IDID for BP.

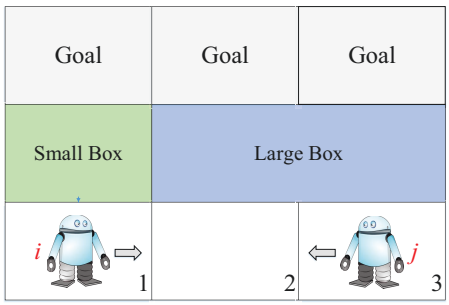

Fig. 9: An illustration of the box-pushing problem domain with two agents: i at position 1 facing *east*, and j at position 3 facing *west* in a 3×3 grid. We denote this state by 1E3W.

Multi-access broadcast channel domain (MABC). In the MABC problem, nodes need to broadcast messages to each other over a channel. Only one node may broadcast at a time, otherwise a collision occurs. The nodes share a common goal of maximizing the throughput of the channel. At the start of each time step, each node can do one of 2 actions: send (S) a

message, or wait (W). We consider a setting where transmissions are assumed to be successful only 90% of the time. The node receives a reward of 1 upon successful transmission of a message and a reward of 0 otherwise. After performing an action, the node receives one of 2 noisy observations: collision (C), or no-collision (NC). The accuracy of these observations is also 90%. The physical state represents the status of the message buffer (whose size = 1) in each node. We illustrate the two-agent MABC problem in Fig. 10 and the I-DID in the Appendix.

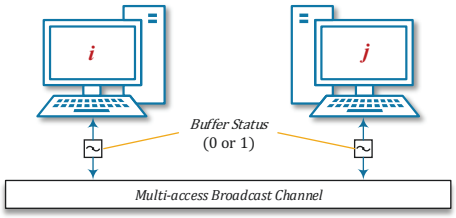

Fig. 10: An illustration of the MABC problem domain with two agents, i and j, each with a buffer size of 1. The status of the buffer of each agent indicates whether a message is ready to be sent or received. For example, a *collision* will occur if both agents' *buffer status* is 1 and one of them tries to send a message over the multiacess channel to the other. We denote the joint status as 11.

Domain	T	$\lvert\mathcal{M}_j^0\rvert$	$\lvert\hat{\Pi}_i\rvert$	Dimension
MABC	3	100	32	$\lvert S_j\rvert$=2, $\lvert S_i\rvert$=4, $\lvert\Omega\rvert$=2, $\lvert A\rvert$=2
	4	100	64	
	5	200	64	
BP	3	100	32	$\lvert S\rvert$=50, $\lvert\Omega\rvert$=5, $\lvert A\rvert$=4
	4	100	64	
Grid	3	100	32	$\lvert S_j\rvert$=9, $\lvert S_i\rvert$=81, $\lvert\Omega\rvert$=3, $\lvert A\rvert$=5
	4	200	100	
4x4Grid	3	100	32	$\lvert S_j\rvert$=16, $\lvert S_i\rvert$=256, $\lvert\Omega\rvert$=3, $\lvert A\rvert$=5
	4	200	100	
6x6Grid	3	100	32	$\lvert S_j\rvert$=36, $\lvert S_i\rvert$=1296, $\lvert\Omega\rvert$=3, $\lvert A\rvert$=5
	4	200	100	

Table 1: Domain dimensions and experimental configuration.

We summarize the properties of the three domains and parameter settings of the Aug. I-DID in Table 1. Note that $|\mathcal{M}_j^0|$ is the number of initial models of agent j at level 0 inclusive of both learning and non-learning models, and $\hat{\Pi}_i$ is the subset of i's policies generated using the principled hill-climbing approach mentioned earlier, allowing us to reduce the full space of j's policies to those that are possibly collaborative. These were obtained from

multiple restarts of the iterative hill climbing procedure. The median number of iterations until termination were 3 with some taking as many as 6 iterations for the Grid domain. Additional experimental settings for each phase are presented in corresponding subsections that follow.

5.2 Teamwork in Finitely-Nested I-DIDs

We implemented the algorithm AUGMENTED I-DID as shown in Fig. 8 including an implementation of the generalized MCESP for performing level 0 RL.

5.2.1 Policy Illustrations

For establishing correctness, we first illustrate the policies obtained using Augmented I-DIDs and compare them with those obtained from a DEC-POMDP formulation of the Grid and BP problem domains.

Grid We begin our illustration with the 3×3 Grid and formulate a level 1 augmented I-DID for agent i. Agent i believes with probability 1 that it is located in the middle cell (cell 4) in the grid, and believes that j believes with probability 1 that it is also located in the middle cell. Additionally, agent i assigns all its probability mass to a single learning model $m'_{j,0}$ of the other agent with $\alpha = 0.8$ and whose seed policy permits moderate exploration. We may formulate an analogous level-1 I-DID for agent j as well. Both agents face a $T = 3$ step planning problem.

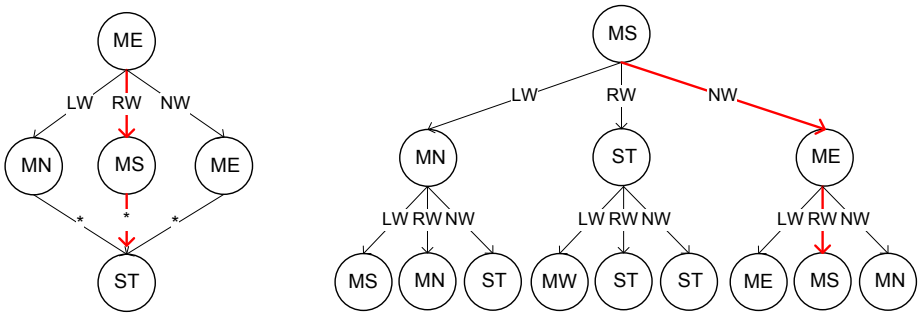

Fig. 11: Policy trees for agent j obtained from RL FOR LEVEL 0 MODEL method. Learning predominantly results in policy tree (a) with the policy in (b) appearing as well. Paths leading to team behavior are highlighted in red in each policy.

Solving the augmented I-DID recursively invokes RL FOR LEVEL 0 MODEL, which requires the policy for agent i as input in order to correctly simulate the grid meeting environment and its team rewards for the learning to take place. In this demonstration of correctness and to speed up agent j's learning process, we let agent i's input policy be that of a teammate.

We show the learned policy tree for agent j using the generalized MCESP in Fig. 11(a). Generalized MCESP predominantly converges to this policy in about 10,000 iterations of

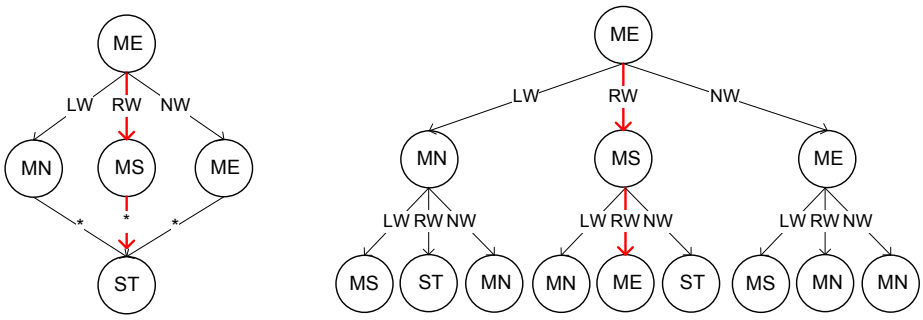

Fig. 12: Solutions of the level 1 I-DID for agent i given the learned policies of j shown in Fig. 11. Paths resulting in team behavior are highlighted in red.

the outer loop in Fig. 6. For those cases where this policy is not learned, the algorithm converges to the policy shown in Fig. 11 (b). In response to the level-0 agent j learning each of these policies, the corresponding level 1 I-DID's solution is shown in Figs. 12(a) and (b), respectively.

Policies for agents i and j as shown in Figs. 11(a) and 12(a) are identical and represent the optimal team behavior for the grid meeting problem when both agents believe that they are situated in the middle cell. [2] In particular, these joint policies have an expected value of 35.97 and are identical to the Pareto-optimal joint policies obtained by running GMAA*-ICE – a recognized algorithm for solving DEC-POMDPs exactly – from the Multiagent Decision Process toolbox [66]. As we highlight in the two policies, both agents first move east and then move down on observing a wall on their right thereby reaching the goal in the bottom-right corner. Both agents then choose to stay in the goal position in the final step after they observe a wall on their right again.

Policies for agents i and j as shown in Figs. 11(b) and 12(b) involve the agents reaching the goal through different routes. Here, agent i first moves east and then moves down on observing a wall on its right. Meanwhile, agent j moves down instead, in the first step, and then moves east when it observes no walls on either side. Both agents ultimately reach the goal (the bottom-right cell). This policy does not involve traveling together and therefore is of lesser value, 33.89, compared to the previous joint policy, but the difference is small.

BP To illustrate, we again formulate a level 1 augmented I-DID for agent i. Agent i believes with probability 1 that the joint physical state of the problem is 1E3W, which denotes i at position 1 and facing east, and j at position 3 facing west as shown in Fig. 9. Agent i believes that j also assigns probability 1 to this physical state. Additionally, agent i assigns all its probability mass to a single learning model, $m'_{j,0}$, of the other agent with $\alpha = 0.8$ and whose seed policy permits moderate exploration as in the previous domain. We may formulate an analogous level-1 I-DID for agent j as well. We let both agents face $T = 3$ step and 4 step planning problems.

[2] Policy shown in Fig. 11(a) is also obtained when agent j is modeled using a level 1 I-DID, and models i at level 0.

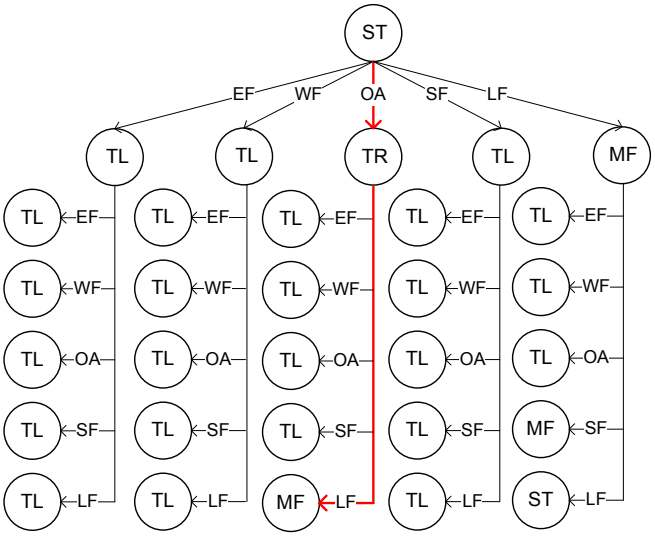

Fig. 13: One of the two learned policies for the level 0 agent j in the box pushing problem.

In this demonstration of correctness on BP and to speed up agent j's learning process in a domain that is large, we let agent i's input policy to level 0 RL be that of a teammate. As we mentioned previously, we plan to experiment with arbitrarily crafted policies that contain exploration thereby allowing the teammate behavior to occur with a non-zero probability.

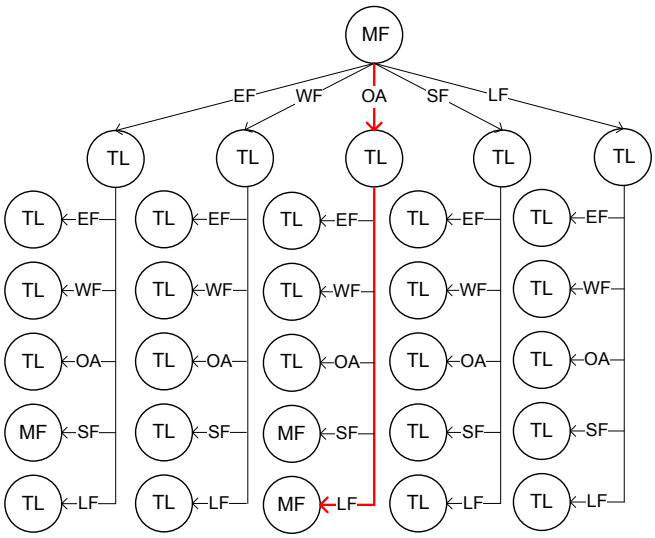

Fig. 14: Solution of the level-1 I-DID of agent i given the learned policy of j in Fig. 13.

We show one of the horizon-3 learned policy for the level 0 j in Fig. 13. These policies were learned in approximately 10,000 iterations of the outer loop of the level 0 RL algorithm. We expect j to behave according to the highlighted path in the tree. Consequently, j begins by staying and on observing the other agent, it turns right thereby facing the large box. This is followed by moving forward and pushing the box. The other learned policy differs from this one in the order of the first two actions, which prescribe j moving right first followed by staying.

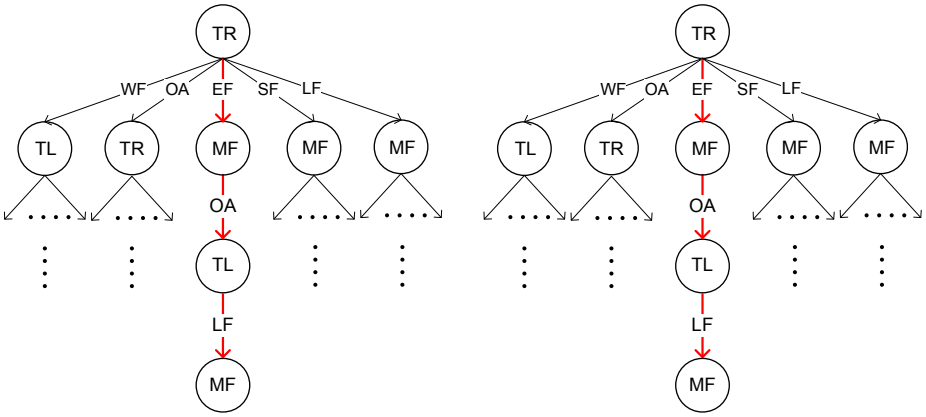

Fig. 15: (a) 4-step learned policy for j, and (b) solution of the level 1 I-DID for i for the box pushing problem when i faces north and j remains unchanged in Fig. 9.

In response to either of these two learned policies of j, solution of the level 1 I-DID generates the policy shown in Fig. 14. This prescribes that agent i move forward and on observing another agent turn left followed by moving forward thereby pushing the large box when it senses it. Notice that when we combine agent i's policy with any of j's two learned policies, the two agents would likely behave according to the highlighted paths. This behavior is the optimal team behavior in this problem domain with the given initial beliefs, as verified by the Pareto-optimal output of the GMAA*-ICE algorithm from the Multiagent Decision Process Toolbox.

We additionally experimented with an increased planning horizon of 4 time steps in the box-pushing problem. Specifically, agent i believes that the physical state is 1N3E and that j knows this as well. We show the policy learned after about 50,000 iterations for j in Fig. 15(a). Because the entire policy tree is too large to show here, we focus on a key path in the policy. Level 1 I-DID's solution given this learned policy is shown in Fig. 15(b). Note that the joint policies represent the optimal behavior for the particular start state, and the highlighted paths in both policies is the optimal teamwork that will likely occur.

5.2.2 Expected Utility Comparisons

Next, we compare the expected utility of agent i's policies with the values of the optimal team policies obtained using an exact algorithm, GMAA*-ICE, for DEC-POMDP formula-

tions of all three problem domains [66]. We also compare with the values obtained by traditional I-DIDs. All I-DIDs are solved using the exact discriminative model update (DMU) method [75], which is the state of the art. For both traditional and augmented I-DIDs, we utilized $|\mathcal{M}_j^0|$ models at level 0 that differ in the initial beliefs or in the frame. We adopt two model weighting schemes as the prior distribution over models: (a) **Uniform**: all policies are uniformly weighted; (b) **Proportional**: let policies with larger expected utility be assigned proportionally larger weights. Note that we maintain the top K by expected utility (out of $|\mathcal{M}_j^0|$) learning and non-learning models only while solving Augmented I-DIDs. Though the model space is significantly enlarged by the learning policies, the tractability of Augmented I-DIDs improves significantly when both top-K and equivalence techniques are applied.

In Table 2, we observe that the Augmented I-DID significantly outperforms the traditional I-DID when level-0 agent j may learn, in all domains MABC, BP, and Grid; we scaled the latter to larger 6×6 grid sizes. The traditional I-DID is solved using the I-DID Exact algorithm of Fig. 4. Augmented I-DID's solutions approach the globally optimal team behavior as generated by GMAA*-ICE. We observe that the larger weights on the learned policies lead to better quality i's policies. This restates the importance of level-0 models that perform RL. The small gap from the optimal DEC-POMDP value is due to the uncertainty over different models of j. Note that DEC-POMDPs are informed about initial beliefs of all agents (and do not face the issue of bounded rationality) whereas I-DIDs are not and they consider the entire candidate model space of j. *Furthermore, the augmented I-DID generates the optimal team behavior identical to that provided by GMAA*-ICE when i's belief places probability 1 on the true model of j, as is the setting for DEC-POMDPs.* Increasing K does not have a significant impact on the performance as K is large enough to cover a large fraction of collaborative policies of agent j including that of the optimal teammate.

In Fig. 16, we illustrate the reduction of model space that occurs due to smaller values of K, which facilitates efficiency in the solution of the augmented I-DID. Though the augmented level-0 model space is much larger than that of its traditional counterparts, the growth in the number of models is limited due to the top-K heuristic.

5.3 Application to Ad Hoc Teams

Having established that augmented I-DIDs can yield teammate plans in a variety of domains, we evaluate their applicability to ad hoc teams, which involve multiple agents interacting without prior coordination. We describe the settings next followed by reporting on the performance evaluation.

5.3.1 Experimental Settings

We test the performance of augmented I-DIDs in ad hoc teamwork applications involving different teammate types and compare it with a known ad hoc planner, OPAT [74]. Teammate types are similar to those used in OPAT and include: (a) **Random** - when the teammate plays according to a randomly generated action sequence for the entire length of the run. A predefined set of random seeds are used to guarantee that each test has the same action sequences. (b) **Predefined** - when the teammate plays according to some predefined patterns; these are sequences of random actions each repeated some number of times that is randomly chosen at the beginning of each run. For example, if the action pattern is"1324" and the repetition value is 2, the resulting action sequence will be "11332244". (c) **Optimal** - when the

	Augmented I-DID			Traditional I-DID
Domain	**K**	**Uniform**	**Proportional**	**Uniform**
MABC **(T=3)**	32	2.12	2.30	1.79
	16	2.12	2.30	
	8	2.12	2.30	
	DEC-POMDP (GMAA*-ICE): 2.99			
MABC **(T=4)**	64	3.13	3.17	2.80
	32	3.13	3.17	
	16	3.13	3.17	
	DEC-POMDP (GMAA*-ICE): 3.89			
MABC **(T=5)**	64	4.08	4.16	3.29
	32	3.99	4.16	
	16	3.99	4.16	
	DEC-POMDP (GMAA*-ICE): 4.79			
BP **(T=3)**	32	73.45	76.51	4.75
	16	73.45	76.51	
	8	71.36	76.51	
	DEC-POMDP (GMAA*-ICE): 85.18			
Grid **(T=3)**	32	41.875	41.93	25.70
	16	40.95	41.93	
	8	40.95	41.93	
	DEC-POMDP (GMAA*-ICE): 43.86			
Grid **(T=4)**	100	37.15	53.26	21.55
	64	35.33	53.26	
	32	35.33	53.26	
	DEC-POMDP (GMAA*-ICE): 58.75			
4x4Grid **(T=3)**	32	29.25	29.60	19.82
	16	29.25	29.60	
	8	27.33	29.60	
	DEC-POMDP (GMAA*-ICE): 31.37			
4x4Grid **(T=4)**	100	33.58	44.15	24.22
	64	33.58	44.15	
	32	31.05	44.15	
	DEC-POMDP (GMAA*-ICE): 49.27			
6x6Grid **(T=3)**	32	46.27	48.35	31.50
	16	46.27	48.35	
	8	46.27	48.35	
	DEC-POMDP (GMAA*-ICE): 55.16			
6x6Grid **(T=4)**	100	60.01	64.18	41.68
	64	60.01	64.18	
	32	55.33	64.18	
	DEC-POMDP (GMAA*-ICE): 69.27			

Table 2: Performance comparison between the traditional I-DID, Augmented I-DID, and GMAA*-ICE in terms of the expected utility.

teammate plays rationally and adaptively. OPAT uses an optimal teammate policy for simulations, which is computed offline with the help of a generative model by value iteration. Note that OPAT in its original form assumes perfect observability of the state and joint ac-

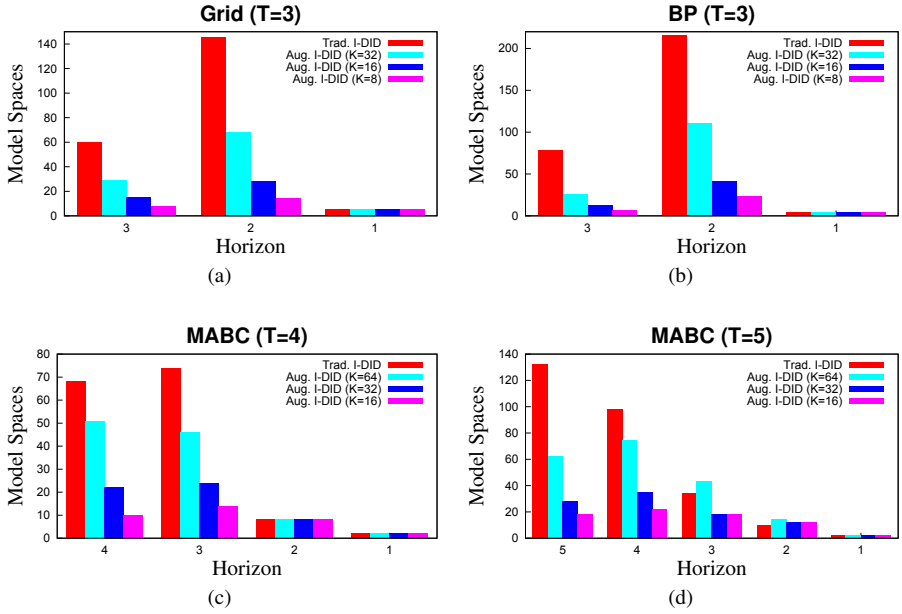

Fig. 16: Top-K method reduces the added solution complexity of the augmented I-DID. The complexity is dominated by the model space (number of models) in each time step.

tions. For comparison, we generalized OPAT to partially observable settings by considering observation sequences.

Additionally, in order to speed up the generation of RL models at level 0, we implemented an approximate version of our generalized MCESP called the Sample Average Approximation that estimates action values by taking a fixed number of sample trajectories and then comparing the sample averages [59]. We used a sample size of n = 25 trajectories to compute the approximate value of the policy that generated them. We set α = 0.9, and terminate the RL (line 15 in Fig. 6) if no policy changes are recommended after taking n samples of the value of each observation sequence-action pair [59]. We also tested with some domain-specific seed policies to investigate speedup in the convergence of MCESP.

Simulations were run for 20 steps and the average of the cumulative rewards over 10 trials are reported for similar teammate settings for the 3 problems. We show that the augmented I-DID solution *significantly* outperforms OPAT solutions in all problem domains for random and predefined teammates while performing comparably for optimal ones.

5.3.2 Performance Evaluation

Next, we compare with OPAT when the true model is included in the candidate model space, and for evaluating robustness, the true model is not in the model space.

Table 3 shows that I-DIDs significantly outperform OPAT especially when the other agents are *random* or *predefined* types in all three problem domains (Student's t-test, p-value ≤ 0.001 for both) except when the teammate is of type *predefined* in MABC where the

Ad Hoc Teammate	OPAT	Augmented I-DID	
		True model excluded	True model included
Grid *T=20, look-ahead=3*			
Random	12.25 ± 1.26	14.2 ± 0.84	–
Predefined	11.7 ± 1.63	16.85 ± 1.35	–
Optimal	28.35 ± 2.4	26.83 ± 3.45	27.96 ± 1.92
BP *T=20, look-ahead=3*			
Random	29.26 ± 2.17	36.15 ± 1.95	–
Predefined	41.1 ± 1.55	54.43 ± 3.38	–
Optimal	52.11 ± 0.48	56.12 ± 3.13	59.2 ± 1.55
MABC *T=20, look-ahead=3*			
Random	9.68 ± 1.37	12.13 ± 1.08	–
Predefined	12.8 ± 0.65	13.22 ± 0.21	–
Optimal	16.64 ± 0.28	14.23 ± 0.88	15.97 ± 1.31

Table 3: Baseline comparison with OPAT in the context of different types of teammates and model spaces. For robustness, we experiment in settings where the true teammate model is included in and excluded from the Augmented I-DID's candidate model space[3]. Each data point is the average of 10 runs and standard deviations are shown. '–' denotes a setting that is not applicable.

improvement over OPAT was not significant at the 0.05 level (p-value = 0.0676). Augmented I-DID's better performance is in part due to the sophisticated Bayesian belief update that gradually increases the probability on the model that best explains the observations (true model if it is present) in agent j's model space as shown in Fig. 17 for MABC.

As expected, both OPAT and augmented I-DID based ad hoc agent perform better when the other agent in the team is optimal in comparison to *random* or *predefined* type. Augmented I-DIDs perform significantly better than OPAT when faced with optimal teammates in BP, while the results for the other domains are similar. We also note that augmented I-DIDs perform significantly (at the 0.05 level) better when the true model of the teammate was included in the ad hoc agent's candidate model space than when it was not except in the Grid domain (p-value = 0.058) where the result was not significant at the 0.05 level.

In summary, the augmented I-DID maintains a probability distribution over different types of teammates and updates both the distribution and types over time, which differs from OPAT's focus on a single optimal behavior of teammates during planning. Consequently, augmented I-DIDs allow better adaptivity as examined above. Further experiments on the robustness of augmented I-DIDs in dynamic ad hoc settings showed that agent i obtained significantly better average rewards compared to OPAT (p-value = 0.042) for the setting where the other agent is of type *predefined* and after 15 steps switches to an *optimal* type for the remaining 15 steps in MABC.

In Fig. 18, we show the online run times for the augmented I-DID and generalized OPAT approaches on the three problem domains. Expectedly, OPAT takes significantly less time

[3] With *random* or *predefined* teammate types, due to the randomness involved in generating their behaviors, the true model of the teammate may not be present in the subject agent's candidate model space.

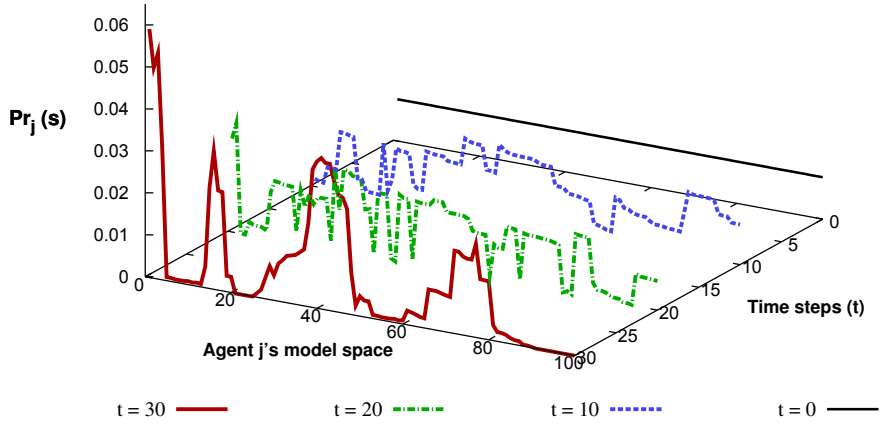

Fig. 17: Bayesian belief updates over 30 steps in MABC showing the beliefs converging to fewer models (largest belief is that of agent j's true model).

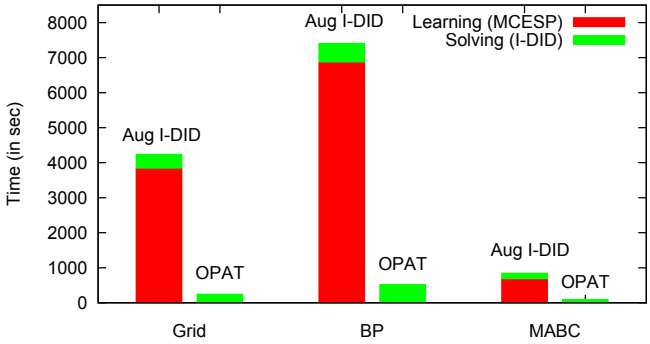

Fig. 18: Timing results for augmented I-DID simulations and OPAT with an OPTIMAL teammate.

because it approximates the problem by solving a series of stage games modeling the other agent using a single type. In the case of augmented I-DIDs, we observe that generating and solving the added learning models consume the major portion of the total time. We show the learning overhead for Grid, BP, and the MABC in red in the figure. To reduce this overhead and speed up augmented I-DIDs, an avenue for future work is to relax the fully model free RL of MCESP and find a middle ground in using some aspects of the model.

5.4 Scalability

Although we recognize that the learning component (MCESP) is the bottleneck to scaling augmented I-DIDs for larger problems, we were successful in obtaining the optimal teammate policies using augmented I-DIDs (same as those computed by **GMAA*-ICE**) in the 4×4 Grid (256 physical states) and 6×6 Grid (1,296 physical states) for $T = 3$ and

$T = 4$, BP for $T = 4$, and MABC for $T = 5$. For these larger problems, we also noticed a significant improvement in the values obtained by augmented I-DIDs over their traditional counterparts as shown in Table 2. Semi-model based learning in MCESP and other approximation techniques, will allow us to further scale-up augmented I-DIDs. Furthermore, by exploiting symmetry and other structural properties of applicable pragmatic domains, the effective complexity of planning may be greatly reduced.

6 Concluding Remarks

Self-interested individual decision makers face hierarchical (or nested) belief systems in their multiagent planning problems. In this article, we explicate one negative consequence of bounded rationality: the agent may not behave as an optimal teammate. In the I-DID framework that models individual decision makers who model other agents, possibly recursively, we show that reinforcement learning integrated with the planning allows the models to produce sophisticated policies. For the first time, we see the principled and comprehensive emergence of team behavior in I-DID solutions facilitating I-DIDs' application to ad hoc teamwork. This is of significance because I-DIDs are naturally well-suited for spontaneous interactions with agents of unknown type. We show that integrating learning in the context of I-DIDs helps us provide a solution to fundamental challenges in ad hoc teamwork – building a single autonomous agent that can plan individually in partially observable environments by adapting to different kinds of teammates while making no assumptions about its teammates' behavior or beliefs and seeking to converge to their true types. Augmented I-DIDs compare well with a standard baseline algorithm, OPAT.

While individual decision-making frameworks such as I-POMDPs and I-DIDs are thought to be well suited for non-cooperative domains, we show that they may be applied to cooperative domains as well. Integrating learning while planning provides a middle ground (or a bridge) between multiagent planning frameworks such as DEC-POMDPs and joint learning for cooperative domains [58]. Spaan *et al.* [67] demonstrate a similar use of learning to get the communication started in each agent's policy in a DEC-POMDP. This is needed because agents will not choose to send messages as the other agent has not yet learned how to assimilate messages in its planning. Agents are expected to learn when and how to communicate as well as how to interpret the communication. Augmented I-DIDs differentiate themselves from other centralized cooperative frameworks by focusing on the behavior of an individual agent in a multiagent setting.

While we recognize that introducing learning-based models adds a significant challenge to scaling I-DIDs for larger problems, we successfully obtained optimal teammate policies using augmented I-DIDs in the 6×6 Grid and BP using a combination of intuitive pruning techniques. By allowing models formalized as I-DIDs or DIDs to vary in the beliefs and frames, we considered an exhaustive and general space of models during planning. The convergence of RL is not predicated on any prior assumptions about other's models. Additionally, problem-specific solutions that address the inadequacy of traditional I-DIDs cannot be ruled out. However, this article contributes a general and principled way to address the inadequacy that should benefit researchers working in various domains; this makes for a stronger contribution.

Immediate lines of future work involve improving the scalability of the framework, particularly its learning component by exploring semi-model based approaches to learning, and investigating much larger pragmatic domains which have structural properties (like symmetry) that can be exploited for tractability [40,41,55].

7 Appendix

We show I-DIDs for three problem domains – BP, Grid, and MABC. For the sake of clarity, we limit the illustration to two-agent settings in which the level-1 agent i considers two level-0 models for the other agent j. The two IDs differ in their beliefs over the physical states. The conditional probability distributions (CPDs) of all the nodes in each I-DID are specified according to the problem described earlier in Section 5.

7.1 Multiagent Box Pushing

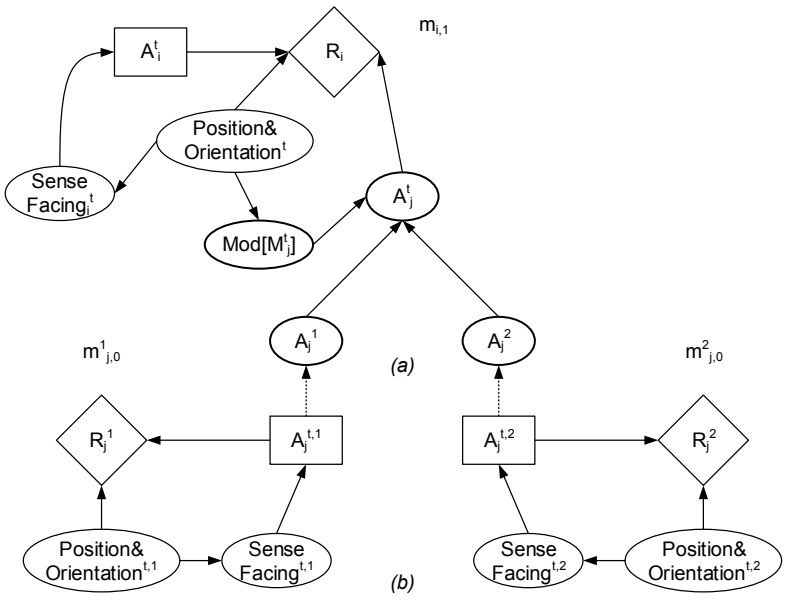

Fig. 19: (a) Level-1 I-ID of agent i in the multiagent BP domain, (b) two level-0 IDs of agent j whose decision nodes are mapped to the chance nodes, A_j^1, A_j^2, in (a), indicated by the dotted arrows. The two IDs differ in their distributions over the chance node, *Position&Orientation*

First, an abstract representation of the level-1 I-ID for BP is shown in Fig. 19. The physical state specifies the joint position and orientation of both the agents. We represent this composite state space by a chance node labeled *Position&Orientation*. Each agent may sense the presence of a wall, other agent, a box, or an empty field in the direction it is facing. These observations are modeled by another chance node *SenseFacing*. We may unroll the I-ID in Fig. 19 into an I-DID spanning two time slices as shown in Fig.20. The model node, $M^t_{j,0}$, contains the different DIDs that are expanded from the level-0 IDs in Fig. 19(b).

We may further exploit the structure of the problem by factoring the state space into *position* and *orientation* indexed by each participating agent. We draw additional benefits

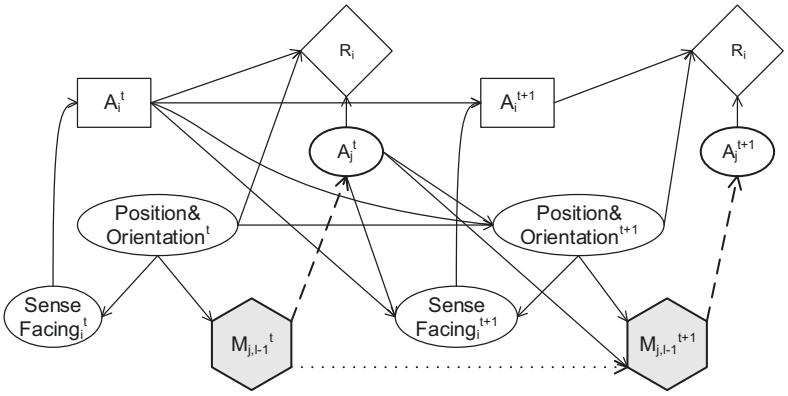

Fig. 20: An abstract two time-slice level 1 I-DID for agent i in BP. The model node contains level 0 DIDs of agent j that expand the IDs.

from also factoring the action space because some actions only impact certain factors of the state. For example, each agent may choose to perform one of 4 possible actions – turn left (TL), turn right (TR), move forward (MF) and stay (ST). The *turn* actions impact the *orientation* of the agent only, while the *move* and *stay* actions impact the agent's *position* in the grid only. We illustrate this factorization of the chance nodes *Position&Orientation* and A_j, and the decision node A_i, in Fig. 21.

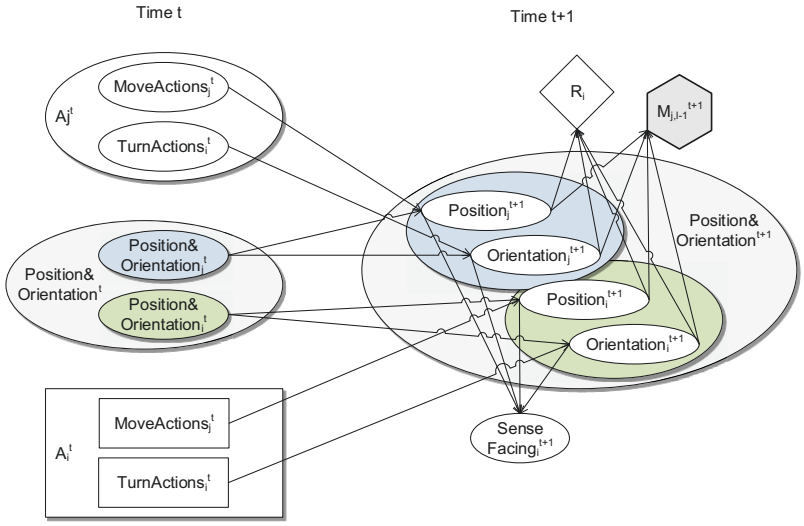

Fig. 21: *Position* and *Orientation*: Per-agent factors of the composite state space in BP abstractly represented by the node *Position&Orientation*; *TurnActions* and *MoveActions*: Per-agent factors of the composite action space modeled by nodes A_i and A_j.

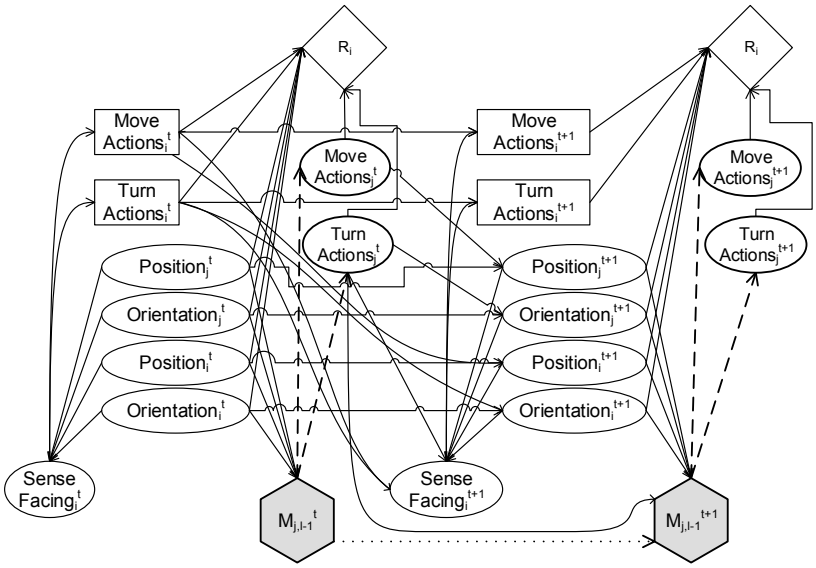

Fig. 22: A factored representation of the two time-slice level 1 I-DID for agent i in BP.

Finally, in Fig. 22, we illustrate a fully factored representation of the level-1 I-DID (shown in Fig. 20) for agent i in BP.

7.2 Multiagent Grid Domain

An abstract representation of the level-1 I-ID for Grid is shown in Fig. 23. The physical states in this domain represent the joint location (x, y coordinates) of each agent in the grid. This composite space is modeled by the chance node *GridLocation*. We may unroll the I-ID in Fig. 23 into the corresponding I-DID spanning two time slices as shown in Fig. 24.

In agent i's I-DID, we assign the marginal distribution over the agents' joint location to the conditional probability distribution (CPD) of the chance node $\textit{GridLocation}_i^t$. In the next time step, the CPD of the chance node $\textit{GridLocation}_i^{t+1}$, conditioned on $\textit{GridLocation}_i^t$, A_i^t, and A_j^t, is the transition function. The CPDs of the chance node $\textit{GridLocation}_i^{t+1}$, the observation node $\textit{SenseWall}_i^{t+1}$, and the utility node R_i are specified according to the problem described in Section 5. Finally, the CPD of the chance node $Mod[M_j^{t+1}]$ in the model node, $M_{j,l1}^{t+1}$, reflects which prior model, action and observation of j results in a model contained in the model node.

As in BP, we may factorize the physical state of Grid to specify the agents' corresponding locations in terms of their x and y coordinates, as modeled by the chance nodes *GridX* and *GridY* shown in Fig. 25. On performing the action(s) at time step t, j may receive observations that detect the presence of a wall on its right, left, or the absence of it on both sides, as modeled in the observation node *SenseWall*. This is reflected in new beliefs on agent j's position in the grid within j's DIDs at time step $t+1$. Consequently, the model node, $M_{j,0}^{t+1}$, contains more models of j and i's updated belief on j's possible DIDs.

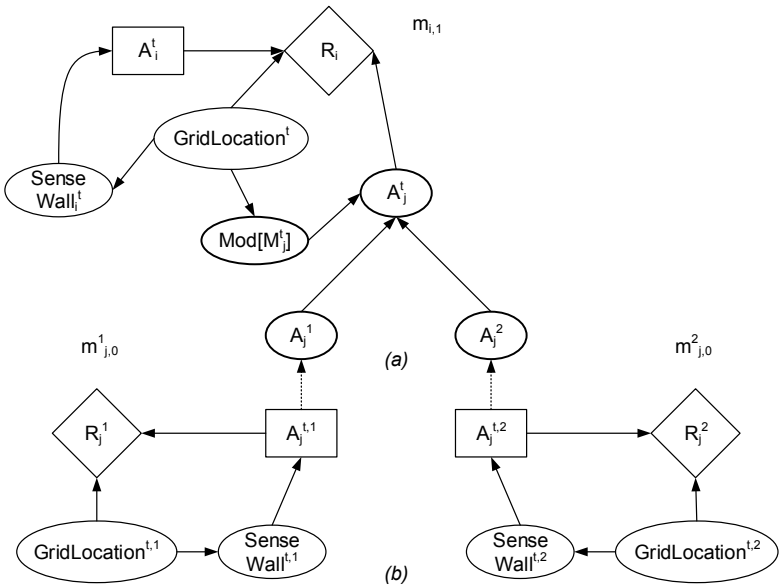

Fig. 23: (a) Level 1 I-ID of agent i in the multiagent Grid domain, (b) two level 0 IDs of agent j whose decision nodes are mapped to the chance nodes, A_j^1, A_j^2, in (a), indicated by the dotted arrows. The two IDs differ in the distribution over the chance node, *GridLocation*

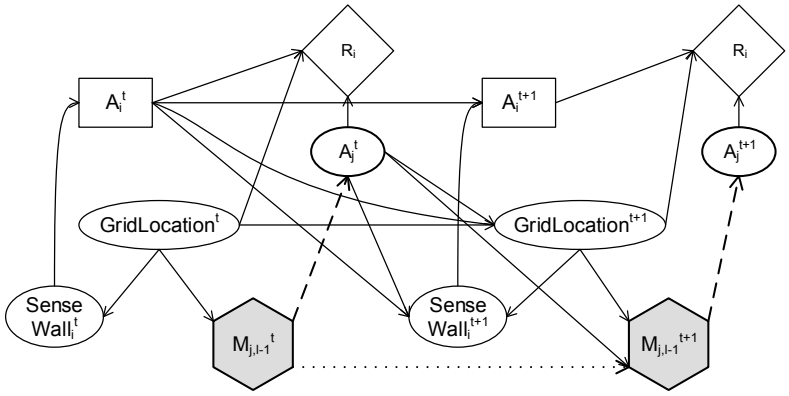

Fig. 24: An abstract two time-slice level-1 I-DID for agent i in Grid. The model node contains level 0 DIDs of agent j. At horizon 1, the models of j are IDs.

Figure 26 illustrates the fully factored representation of the level-1 agent i's I-DID (in Fig. 24) for Grid.

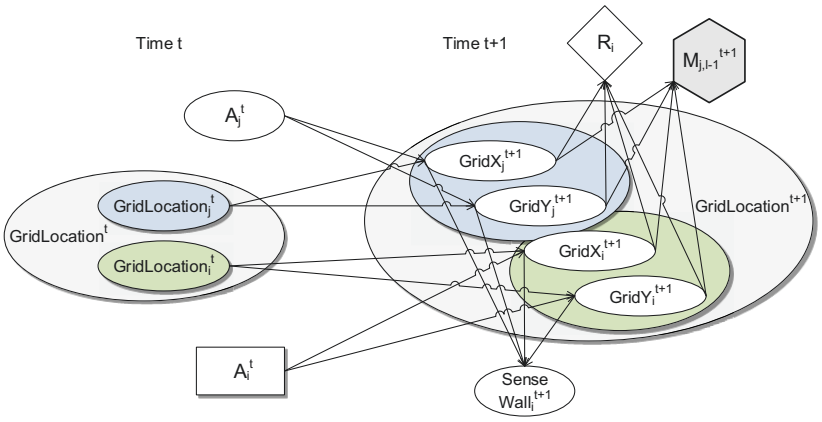

Fig. 25: *GridX* and *GridY*: Per-agent factors of the composite state space in Grid abstractly represented by the node *GridLocation*.

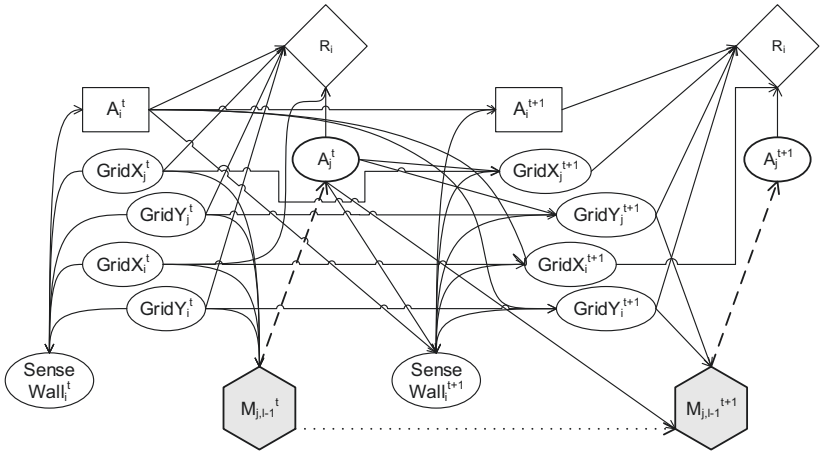

Fig. 26: A fully factored representation of the two time-slice level 1 I-DID for agent i in the multiagent Grid domain.

7.3 Multi-access Broadcast Channel

A representation of the level-1 I-ID for MABC is shown in Fig. 27. The physical state represents the status of the each agent's (i.e., node's) message buffer, whose size is assumed to be 1 in our setting. At the start of each time step, each node performs one of two actions: send a message (S) or wait (W). After performing an action, the node receives one of two noisy observations: collision (C) or no-collision (NC), as modeled by the chance node, *SenseCollision*. We may unroll the I-ID in Fig. 27 into the corresponding I-DID spanning two time-slices as shown in Fig. 28.

In agent i's I-DID, we assign the marginal distribution over the agents' joint buffer status to the CPD of the chance node $BufferStatus_i^t$. In the next time step, the CPD of *Buffer-*

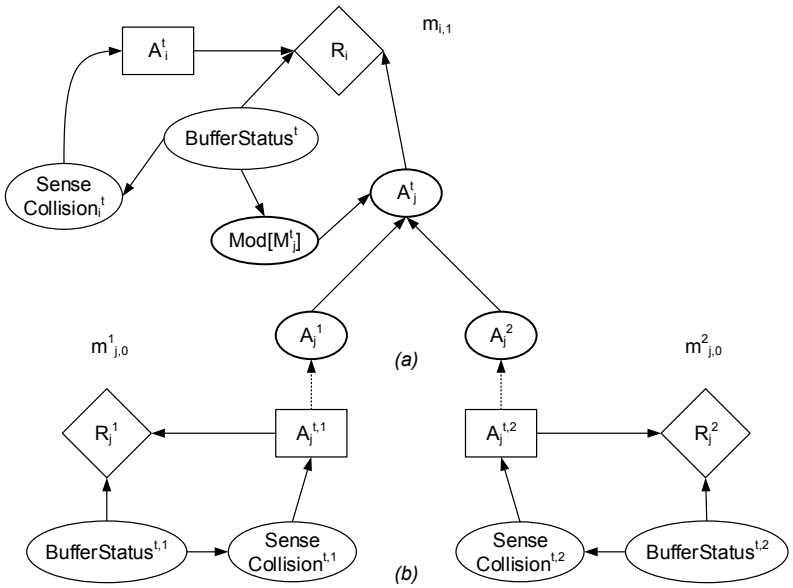

Fig. 27: (a) Level 1 I-ID of agent i in the MABC domain, (b) two level 0 IDs of agent j whose decision nodes are mapped to the chance nodes, A_j^1, A_j^2, in (a), indicated by the dotted arrows. The two IDs differ in the distribution over the chance node, *BufferStatus*

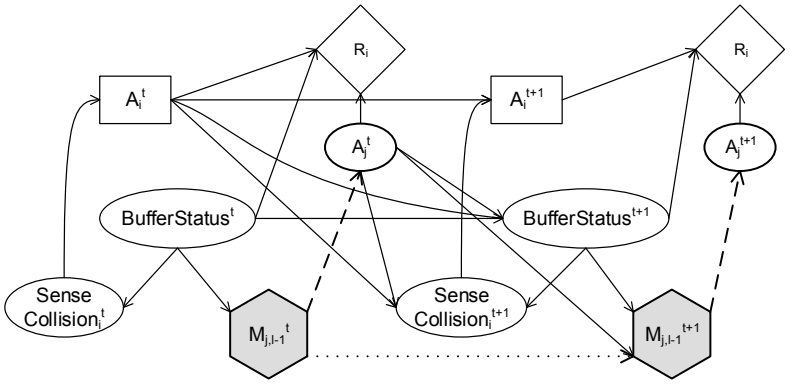

Fig. 28: A two time-slice level 1 I-DID for agent i in the MABC domain. The model node contains level-0 DIDs of agent j that expand the IDs shown in Fig. $27(b)$.

Status$_i^{t+1}$, is the transition function. The CPDs of the chance nodes *BufferStatus*$_i^{t+1}$, the observation node *SenseCollision*$_i^{t+1}$, and the utility node R_i are specified according to the problem described in Section 5. Finally, the CPD of the chance node $Mod[M_j^{t+1}]$ in the model node, $M_{j,l1}^{t+1}$, reflects which prior model, action and observation of j results in a model contained in the model node.

References

1. Adam, B., Dekel, E.: Hierarchies of beliefs and common knowledge. International Journal of Game Theory (1993)
2. Adoe, F., Chen, Y., Doshi, P.: Fast solving of influence diagrams for multiagent planning on gpu-enabled architectures. In: International Conference on Agents and Artificial Intelligence (ICAART), pp. 183–195 (2015)
3. Agmon, N., Barrett, S., Stone, P.: Modeling uncertainty in leading ad hoc teams. In: Proceedings of the 13th International Conference on Autonomous Agents and Multiagent Systems (AAMAS) (2014)
4. Agmon, N., Stone, P.: Leading ad hoc agents in joint action settings with multiple teammates. In: Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, pp. 341–348. International Foundation for Autonomous Agents and Multiagent Systems (2012)
5. Agogino, A., Turner, K.: Multi-agent reward analysis for learning in noisy domains. In: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, pp. 81–88. ACM (2005)
6. Albrecht, S., Crandall, J., Ramamoorthy, S.: Belief and truth in hypothesised behaviours. Artificial Intelligence (2016)
7. Albrecht, S., Ramamoorthy, S.: A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. Tech. rep., Univ. of Edinburgh (2013)
8. Albrecht, S., Ramamoorthy, S.: A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems (extended abstract). In: AAMAS, pp. 1155–1156 (2013)
9. Albrecht, S., Ramamoorthy, S.: On convergence and optimality of best-response learning with policy types in multiagent systems. In: Proceedings of the 30th Conference on Uncertainty in Artificial Intelligence (UAI-14). Quebec City, Canada (2014)
10. Albrecht, S., Ramamoorthy, S.: Are you doing what i think you are doing? criticising uncertain agent models. In: Proceedings of the 31st Conference on Uncertainty in Artificial Intelligence (UAI-15). Amsterdam, Netherlands (2015)
11. Amato, C., Konidaris, G.D., Kaelbling, L.P.: Planning with macro-actions in decentralized pomdps. In: Proceedings of the 2014 international conference on Autonomous agents and multi-agent systems, pp. 1273–1280. International Foundation for Autonomous Agents and Multiagent Systems (2014)
12. Amato, C., Oliehoek, F.A.: Scalable planning and learning for multiagent pomdps. In: Proceedings of the 29th AAAI Conference on Artificial Intelligence (2015)
13. Aumann, R.J.: Interactive epistemology II: Probability. International Journal of Game Theory **28**, 301–314 (1999)
14. Banerjee, B., Lyle, J., Kraemer, L., Yellamraju, R.: Solving finite horizon decentralized pomdps by distributed reinforcement learning. In: AAMAS Workshop on MSDM, pp. 9–16 (2012)
15. Barrett, S., Stone, P., Kraus, S.: Empirical evaluation of ad hoc teamwork in the pursuit domain. In: AAMAS (2011)
16. Bernstein, D.S., Givan, R., Immerman, N., Zilberstein, S.: The complexity of decentralized control of Markov decision processes. Mathematics of Operations Research **27**(4), 819–840 (2002)
17. Bernstein, D.S., Hansen, E.A., Zilberstein, S.: Bounded policy iteration for decentralized pomdps. In: IJCAI (2005)
18. Binmore, K.: Essays on Foundations of Game Theory. Pittman (1982)
19. Boutilier, C.: Sequential optimality and coordination in multiagent systems. In: IJCAI, vol. 99, pp. 478–485 (1999)
20. Bowling, M., McCracken, P.: Coordination and adaptation in impromptu teams. In: AAAI, vol. 5, pp. 53–58 (2005)
21. Bowling, M.H., McCracken, P.: Coordination and adaptation in impromptu teams. In: AAAI, pp. 53–58 (2005)
22. Brandenburger, A.: The power of paradox: Some recent developments in interactive epistemology. International Journal of Game Theory **35**, 465–492 (2007)
23. Brown, G.W.: Iterative solution of games by fictitious play. Activity analysis of production and allocation **13**(1), 374–376 (1951)
24. Camerer, C.: Behavioral Game Theory: Experiments in Strategic Interaction. Princeton University Press (2003)
25. Camerer, C.F., Ho, T.H., Chong, J.K.: A cognitive hierarchy model of games. The Quarterly Journal of Economics **119**(3), 861–898 (2004)
26. Carlin, A., Zilberstein, S.: Value-based observation compression for dec-pomdps. In: Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 1, pp. 501–508. International Foundation for Autonomous Agents and Multiagent Systems (2008)

27. Chakraborty, D., Stone, P.: Cooperating with a markovian ad hoc teammate. In: Proceedings of the 2013 international conference on Autonomous agents and multi-agent systems, pp. 1085–1092. International Foundation for Autonomous Agents and Multiagent Systems (2013)
28. Chandrasekaran, M., Prashant, D., Zeng, Y.: Approximate solutions of interactive dynamic influence diagrams using epsilon-behavioral equivalence. In: 11th International Symposium on Artificial Intelligence and Mathematics, ISAIM (2010)
29. Chang, Y.h., Ho, T., Kaelbling, L.P.: All learning is local: Multi-agent learning in global reward games. In: Advances in Neural Information Processing Systems, pp. 807–814 (2004)
30. Chrisman, L.: Reinforcement learning with perceptual aliasing: the perceptual distinctions approach. In: AAAI, pp. 183–188 (1992)
31. Dibangoye, J.S., Amato, C., Buffet, O., Charpillet, F.: Optimally solving dec-pomdps as continuous-state mdps. In: Proceedings of the Twenty-Third international joint conference on Artificial Intelligence, pp. 90–96. AAAI Press (2013)
32. Doshi, P.: Decision making in complex mulitiagent contexts: A tale of two frameworks. AI Magazine **4**(33), 82–95 (2012)
33. Doshi, P., Chandrasekaran, M., Zeng, Y.: Epsilon-subjective equivalence of models for interactive dynamic influence diagrams. In: Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on, vol. 2, pp. 165–172. IEEE (2010)
34. Doshi, P., Zeng, Y.: Improved approximation of interactive dynamic influence diagrams using discriminative model updates. In: Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 2, pp. 907–914. International Foundation for Autonomous Agents and Multiagent Systems (2009)
35. Doshi, P., Zeng, Y.: Improved approximation of interactive dynamic influence diagrams using discriminative model updates. In: AAMAS (2009)
36. Doshi, P., Zeng, Y., Chen, Q.: Graphical models for interactive pomdps: Representations and solutions. JAAMAS **18**(3), 376–416 (2009)
37. Gal, Y., Pfeffer, A.: A language for modeling agent's decision-making processes in games. In: AAMAS, pp. 265–272 (2003)
38. Gilboa, I., Schmeidler, D.: A theory of case-based decisions. Cambridge University Press (2001)
39. Gmytrasiewicz, P., Doshi, P.: A framework for sequential planning in multiagent settings. JAIR **24**, 49–79 (2005)
40. Goodwine, B., Antsaklis, P.: Multi-agent compositional stability exploiting system symmetries. Automatica **49**(11), 3158–3166 (2013)
41. Guestrin, C., Koller, D., Parr, R.: Multiagent planning with factored mdps. In: NIPS, vol. 1, pp. 1523–1530 (2001)
42. Hansen, E.A., Bernstein, D.S., Zilberstein, S.: Dynamic programming for partially observable stochastic games. In: AAAI, pp. 709–715 (2004)
43. Harsanyi, J.C.: Games with incomplete information played by bayesian players. Management Science **14**(3), 159–182 (1967)
44. Hoang, T.N., Low, K.H.: Interactive pomdp lite: Towards practical planning to predict and exploit intentions for interacting with self-interested agents. In: IJCAI, pp. 2298–2305 (2013)
45. Kalai, E., Lehrer, E.: Rational learning leads to nash equilibrium. Econometrica **61**(5), 1019–1045 (1993)
46. Kim, Y., Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Exploiting locality of interaction in networked distributed POMDPs. In: AAAI Spring Symposium on Distributed Plan and Schedule Management (2006)
47. Koller, D., Milch, B.: Multi-agent influence diagrams for representing and solving games. In: IJCAI, pp. 1027–1034 (2001)
48. Koller, D., Milch, B.: Multi-agent influence diagrams for representing and solving games. Games and economic behavior **45**(1), 181–221 (2003)
49. Kumar, A., Zilberstein, S., Toussaint, M.: Scalable multiagent planning using probabilistic inference. In: IJCAI (2011)
50. Liu, B., Singh, S., Lewis, R.L., Qin, S.: Optimal rewards in multiagent teams. In: 2012 IEEE International Conference on Development and Learning and Epigenetic Robotics (ICDL), pp. 1–8. IEEE (2012)
51. Mccallum, A.K.: Reinforcement learning with selective perception and hidden state. Ph.D. thesis, University of Rochester (1996)
52. Mertens, J., Zamir, S.: Formulation of bayesian analysis for games with incomplete information. International Journal of Game Theory **14**, 1–29 (1985)
53. Meuleau, N., Peshkin, L., eung Kim, K., Kaelbling, L.P.: Learning finite-state controllers for partially observable environments. In: UAI, pp. 427–436 (1999)
54. Nair, R., Tambe, M., Yokoo, M., Pynadath, D., Marsella, S.: Taming decentralized pomdps: Towards efficient policy computation for multiagent settings. In: IJCAI, pp. 705–711 (2003)

55. Nair, R., Varakantham, P., Tambe, M., Yokoo, M.: Networked distributed pomdps: A synthesis of distributed constraint optimization and pomdps. In: AAAI, vol. 5, pp. 133–139 (2005)
56. Ng, B., Boakye, K., Meyers, C., Wang, A.: Bayes-adaptive interactive pomdps. In: AAAI (2012)
57. Oliehoek, F.A., Spaan, M.T., Amato, C., Whiteson, S.: Incremental clustering and expansion for faster optimal planning in dec-pomdps. Journal of Artificial Intelligence Research pp. 449–509 (2013)
58. Panait, L., Luke, S.: Cooperative multi-agent learning: The state of the art. JAAMAS **11**(3), 387–434 (2005)
59. Perkins, T.J.: Reinforcement learning for pomdps based on action values and stochastic optimization. In: AAAI, pp. 199–204 (2002)
60. Pineau, J., Gordon, G., Thrun, S.: Anytime point-based approximations for large pomdps. Journal of Artificial Intelligence Research pp. 335–380 (2006)
61. Pynadath, D., Marsella, S.: Minimal mental models. In: AAAI, pp. 1038–1044 (2007)
62. Pynadath, D.V., Tambe, M.: The communicative multiagent team decision problem: Analyzing teamwork theories and models. Journal of Artificial Intelligence Research **16**, 389–423 (2002)
63. Rathnasabapathy, B., Doshi, P., Gmytrasiewicz, P.: Exact solutions of interactive pomdps using behavioral equivalence. In: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, pp. 1025–1032. ACM (2006)
64. Seuken, S., Zilberstein, S.: Improved memory-bounded dynamic programming for decentralized pomdps. In: UAI (2007)
65. Seuken, S., Zilberstein, S.: Formal models and algorithms for decentralized decision making under uncertainty. Autonomous Agents and Multi-Agent Systems **17**(2), 190–250 (2008)
66. Spaan, M., Oliehoek, F.: The multiagent decision process toolbox: Software for decision-theoretic planning in multiagent systems. In: AAMAS Workshop on MSDM, pp. 107–121 (2008)
67. Spaan, M.T.J.: Decentralized planning under uncertainty for teams of communicating agents. In: AAMAS, pp. 249–256 (2006)
68. Stone, P., Kaminka, G.A., Kraus, S., Rosenschein, J.S.: Ad hoc autonomous agent teams: Collaboration without pre-coordination. In: AAAI (2010)
69. Stone, P., Kaminka, G.A., Rosenschein, J.S.: Leading a best-response teammate in an ad hoc team. In: Agent-Mediated Electronic Commerce. Designing Trading Strategies and Mechanisms for Electronic Markets, pp. 132–146. Springer (2010)
70. Stone, P., Kraus, S.: To teach or not to teach? decision making under uncertainty in ad hoc teams. In: AAMAS (2010)
71. Tatman, J.A., Shachter, R.D.: Dynamic programming and influence diagrams. IEEE Transactions on Systems, Man, and Cybernetics **20**(2), 365–379 (1990)
72. Wageman, R., Baker, G.: Incentives and cooperation: The joint effects of task and reward interdependence on group performance. Journal of organizational behavior **18**(2), 139–158 (1997)
73. Wright, J.R., Leyton-Brown, K.: Level-0 meta-models for predicting human behavior in games. In: Fifteenth ACM Conference on Economics and Computation (EC), pp. 857–874 (2014)
74. Wu, F., Zilberstein, S., Chen, X.: Online planning for ad hoc autonomous agent teams. In: IJCAI, pp. 439–445 (2011)
75. Zeng, Y., Doshi, P.: Exploiting model equivalences for solving interactive dynamic influence diagrams. JAIR **43**, 211–255 (2012)
76. Zeng, Y., Doshi, P., Pan, Y., Mao, H., Chandrasekaran, M., Luo, J.: Utilizing partial policies for identifying equivalence of behavioral models. In: Proceedings of the 25th AAAI Conference on Artificial Intelligence (2011)